Apprentissage d'automates modélisant des familles de séquences protéiques

Goulven Kerbellec

▶ **To cite this version:**

Goulven Kerbellec. Apprentissage d'automates modélisant des familles de séquences protéiques. Interface homme-machine [cs.HC]. Université Rennes 1, 2008. Français. NNT : . tel-00327938

HAL Id: tel-00327938
https://theses.hal.science/tel-00327938

Submitted on 9 Oct 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

N° d'ordre: 3746

THÈSE

Présentée devant

devant l'Université de Rennes 1

pour obtenir

le grade de : Docteur de l'Université de Rennes 1
Mention Informatique

par

Goulven Kerbellec

Équipe d'accueil : projet SYMBIOSE (IRISA)
École Doctorale : Matisse
Composante universitaire : IFSIC

Titre de la thèse :

Apprentissage d'automates modélisant des familles de séquences protéiques

soutenue le 19 juin 2008 devant la commission d'examen

Président :	Laurent	Miclet	Pr. Université de Rennes 1 à l'ENSSAT
Rapporteurs :	Burkhard	Morgenstern	Pr. Universität Göttingen
	Hélène	Touzet	HDR, Chargée de recherche CNRS, LIFL, Lille
Examinateurs :	Rumen	Andonov	Pr. INRIA Rennes - Bretagne Atlantique (directeur)
	François	Coste	CR INRIA Rennes - Bretagne Atlantique (encadrant)
	Joël	Pothier	Maître de conférence, Paris VI

"Si quelqu'un te dit qu'il va faire porter une culotte à un éléphant, s'il le fait, il aura réussi une grande chose. Mais s'il ne le fait pas, il aura dit une grosse bêtise."

Alpha Blondy

Remerciements

La région Bretagne a permis le financement de trois années de cette thèse qui s'est déroulée à l'Irisa au sein de l'équipe Symbiose de l'INRIA Rennes - Bretagne Atlantique.
Comme il m'est donné l'occasion de remercier toutes les personnes qui m'ont permis de réaliser ce parcours particulièrement riche en connaissances à la fois scientifiques et humaines, je commencerai par Jacques Nicolas et l'équipe Symbiose.
C'est une chance d'avoir pu travailler avec des personnes passionnées, soudées, drôles et chaleureuses, à l'image d'un chef d'équipe dont la respectabilité n'a d'égale que le nombre de casquettes/chapeaux/parapluies oubliés.
Merci à Michel pour ses looooongs exposés dont je suis le plus grand fan. Merci à Catherine pour avoir, entre 1000 autres marques d'affection, éliminé le son de son p.c. . Merci aussi à Anne qui a l'acceng et le tempérameng du soleil. Merci à Rumen qui m'a fait l'honneur d'être mon directeur de thèse. Merci encore à Dominique pour sa clairvoyance en toutes circonstances.
Et le grand merci qui va évidemment à François Coste, pour qui il n'a pas dû être facile tous les jours de me gérer, mais avec qui nous passâmes des moments exceptionnels. On a partagé beaucoup de choses ces dernières années : les publis à pas d'heure, les expés interminables, mais surtout les supers protautomates, les voyages, les jeux de mots impossibles, etc.
Merci au groupe des ingés pour leur aide, avec en vrac : Anne-So, Greg, François, Annabel, Gilles, Alex, Anthony, Laëtitia, Sophie, Hugues, Olivier, Julien, Elodie, etc.
Merci à tous mes compagnons de lutte thèsarde : Philippe, Alex, Ronan, Xavier, J.F., Thomas, Daniel, Stéphane, François, Vincent, Ingrid, etc.
Je pense aussi à toute l'aide apportée par les stagiaires : Hanh, Jessie et Hélène. Et puis merci aussi à Marie-Noëlle. Sans oublier tous les dreamers, les pokermen comme Pascal, les basketmen comme Rafik, et les babyfootmen comme Yann et Jeremy.
Mais bon, il n'y avait pas que des informaticiens, il y avait aussi des gens qui m'ont aidé sur la partie biologique. Un grand merci à Christian Delamarche et Thierry Guillaudeux. Et merci aussi à Emeric et sa chouette bande de biologistes.
Merci à toute la famille, particulièrement mes parents et ma soeur, pour la confiance et le soutien qu'ils m'ont apporté. Merci aussi aux amis, de Paimpol ou d'ailleurs, des Kazara aux Banbuck, des Fablet aux Paré. Merci surtout à Patrick de me faire confiance pour la suite.
Et merci enfin à Marie-Laure et à Liam pour leur gentillesse, leurs attentions et surtout d'apporter un sens à ma vie.

Table des matières

Avant-Propos **7**

Introduction **9**

I Etat de l'Art **13**

1 Familles de Protéines **15**
1.1 Repères biologiques 15
1.2 Les modèles 20
1.2.1 Composition en acides aminés 21
1.2.2 Alignement de séquences 22
1.2.3 Alignements multiples 23
1.2.4 Matrices PSSM 24
1.2.5 Profils HMM 25
1.3 Les Motifs courts 26
1.4 Les combinaisons de modèles 28

2 Approche de la théorie des langages **29**
2.1 Théorie des langages 29
2.1.1 Hiérarchie de Chomsky 30
2.1.2 Automates à états finis 31
2.1.3 Expressions régulières 33
2.2 Les séquences génomiques au travers de la théorie des langages 34
2.2.1 Description des données 34
2.2.2 Le langage d'une famille de protéines 35
2.2.3 Expression régulière en bioinformatique 35
2.2.4 Automates en bioinformatique 36
2.3 Apprentissage d'automates d'états finis 36
2.3.1 Introduction 36
2.3.2 Le choix du non déterminisme 37
2.3.3 Algorithmes par fusion d'états 37
2.3.4 Algorithmes d'inférence de DFA et de NFA 40
2.4 Premiers algorithmes d'inférence de NFA appliqués à la biologie 40

2.4.1 Conclusion 41

II Apprentissage d'automates non-déterministes sur des familles de protéines 43

3 Alignement Multiple Partiel et Local 47
3.1 *Partial local multiple alignement* (PLMA) 47
3.2 Similarité 48
3.3 Paires de fragments significativement similaires (SFP) 49
3.4 Evaluation du score d'une SFP 51
3.5 Représentation des SFP sous forme d'un graphe de fragments 51
3.6 Bloc de PLMA 53
3.7 Evaluation du score d'un bloc de PLMA 54
3.8 Construction de blocs de PLMA à partir du graphe des fragments . . . 54
3.9 Incompatibilités entre SFP 55
3.10 Incompatibilités entre blocs de PLMA 57
3.11 Problème : trouver le meilleur PLMA compatible 58
3.12 Algorithmes 58
3.13 Schéma général de l'approche 59
3.14 Résolution par la méthode Strictement SFP 59
3.15 Résolution par méthode Clique Semi-Exhaustive 60
3.16 Résolution par méthode Graines de Blocs de PLMA 61
3.17 Conclusion 62

4 Protomates 65
4.1 Fusion des positions d'un PLMA 65
4.2 Intérêt des automates à transition 67
4.3 Zones caractéristiques 68
4.4 Fusion des états de *gap* 68
4.5 Les exceptions 69
4.6 Propriétés physico-chimiques 73
4.7 Algorithme complet de l'approche 74
4.8 Conclusion générale de l'approche 75

III Evaluation 77

5 Implémentation 81
5.1 Codage 81
5.2 Acceptation d'une séquence dans un modèle 81
5.3 Evaluer le modèle avec une mesure *Minimum Description Lenght* (MDL) 82
5.4 Etude de cas sur le clan Pfam de la super-famille des Viral Nucleic Acid Binding 84

6 Expériences d'Apprentissage de Protomates **89**
6.1 Validation de l'apprentissage 89
6.2 Leave-One-Out 90
6.3 Familles d'intérêt 91
6.3.1 Construction de jeux de données 91
6.3.2 *Major Instrinsic Proteins* (MIP) 92
6.3.3 *Tumor Necrosis Factor* (TNF) 93
6.4 Expérience d'apprentissage d'un motif MIP 95
6.4.1 Résultats 96
6.5 Expérience de discrimination entre deux types de canaux MIP 97
6.5.1 Résultats 98
6.6 Expérience d'apprentissage de signatures de ligands TNF 99
6.7 Etude paramètrique des approches avec graines sur les ligands des TNF 102
6.7.1 Méthode 102
6.7.2 Résultats 103
6.7.3 Conclusion 117
6.8 Courbe MDL 117
6.9 Scan de Swissprot 118
6.10 Conclusion 118

Conclusion **119**
6.11 Avancées obtenues par notre approche 119
6.12 Perspectives 120
6.13 Faut-il aller vers plus d'expressivité ? 121

A Annexe **123**

Bibliographie **124**

Table des figures **131**

Liste des algorithmes **135**

Avant Propos

La problématique à l'origine de cette thèse a été l'étude et l'apprentissage d'un modèle novateur représentant des signatures de familles de protéines.
Ce modèle de type automate non-déterministe encore très peu étudié sur les séquences de protéines était-il capable de faire émerger des propriétés intéressantes sur les familles de protéines ? Était-il aisément apprenable ? Et enfin, était-il un outil suffisamment fin pour retrouver des membres d'une famille dans les banques de données biologiques ?
En raison de son aspect multidisciplinaire, ce rapport s'adresse à des lecteurs issus de plusieurs communautés scientifiques. Les idées, les démarches, et les expériences présentées dans ces travaux représentent alors, en fonction du domaine de compétence du lecteur, des objets de stimulation et de frustration. D'un côté, le lecteur regrettera parfois que certains développements ne soient pas suffisamment approfondis, exploités ou comparés dans le cadre habituel de l'informatique ou de la biologie. D'un autre côté, c'est en avançant de façon transversale dans l'exploration et la combinaison de différentes techniques que nous avons pu montrer l'existence d'une approche fructueuse. Notre stratégie a été exploratoire et guidée vers l'obtention d'un modèle novateur.
Le rôle du bioinformaticien nous semble être de construire des ponts entre les domaines biologiques et informatiques, les bases devant être suffisamment solides pour permettre la rencontre des deux communautés. Chacun devient alors libre de progresser dans un cadre lui assurant la compatibilité avec le domaine opposé.

"Pont de singe" sur la canopée de Bukit Bankirai (source tripalbum.net).

Introduction

Les familles de protéines sont des ensembles de protéines partageant des caractéristiques communes. Ce concept peut sembler simple lorsqu'il s'agit de regrouper sous un même nom plusieurs protéines remplissant des fonctions identiques au sein d'un même organisme ou au travers de différentes espèces. Cependant, au fur et à mesure de l'introduction de nouveaux membres dans une famille de protéines en raison de leur proximité avec certains membres, il devient de plus en plus difficile de faire émerger une propriété unique.

Lorsque l'on étudie avec précision ce que les biologistes nomment famille ou superfamille de séquences, on se rend compte du côté parfois impalpable du concept de famille. Malgré tout, notre objectif est de participer à l'automatisation de la représentation du concept de famille à partir d'échantillons représentatifs de plusieurs protéines membres de cette famille. Il s'agit donc de commencer par étudier d'une part les informations disponibles sur les protéines et leur familles dans diverses banques de données, et d'autre part les différentes approches permettant de générer des signatures caractéristiques de ces familles. Puis il s'agira de montrer en quoi les automates finis non-déterministes semblent être un modèle expressif, efficace et particulièrement novateur de signatures de familles de protéines.

Afin de situer l'état actuel des banques de séquences de protéines, rappelons qu'au 15 janvier 2008, la méta base de données de protéines Uniprot (Bairoch, 2005) annonçait, dans sa version 2.17, le nombre de 5 473 336 entrées. Le nombre de ces entrées a connu une croissance exponentielle depuis le lancement de plusieurs projets de séquençage à grande échelle tels que le HGP (Human Genome Project) entrepris par un consortium international de laboratoire public en 1990. Parmi les entrées d'Uniprot, seulement 333 445 proviennent de la base Swissprot (Boeckmann et coll., 2003), connue pour la fiabilité de ses annotations.

La base de signatures de référence est connue sous le nom d'Interpro (Zdobnov et Apweiler, 2001). Elle liste de nombreuses familles de protéines et y associe des motifs, des profils ainsi que des annotations. Interpro est en fait une méta base de signatures regroupant plusieurs bases telles que Prosite (Hulo et coll., 2004), Prints (Attwood et coll., 2006), Pfam (Bateman et coll., 2004), Panther (Thomas et coll., 2003), CATH (Earl et coll., 2005), etc.

L'obtention d'une signature d'une famille a deux principaux intérêts qu'il s'agit de satisfaire au mieux.

Le premier intérêt réside dans le potentiel de caractérisation, de représentation, et

d'analyse que doit apporter un bon modèle. Le deuxième intérêt tient dans les capacités de prédiction d'appartenance ou non à une famille, à partir de séquences candidates. La bioinformatique connaît de très bon outils prédicteurs tels que les PSSM (Schaffer et coll., 2001), les profils HMM (Eddy, 1998) ou encore les SVM (Guermeur, 2007). Cependant, ces méthodes apparaissent comme des boîtes noires pour l'utilisateur. En effet, l'introduction de raisonnements probabilistes a parfois permis de s'affranchir de l'analyse du contenu des séquences pour n'en retirer que les informations provenant de la fréquence d'apparition des acides aminés à certaines positions. Malheureusement, nous perdons alors les valeurs explicatives et sémantiques d'un modèle discret. En effet, que nous apporte de connaître les occurrences des lettres dans une langue naturelle ? Sûrement un moyen d'en reconnaître aisément des extraits parmi d'autres textes de langues étrangères, mais elles nous apportent très peu d'information sur la construction grammaticale et sur la valeur sémantique des mots et des phrases.
Jusqu'à présent les représentations les plus utilisées par les biologistes pour contenir les connaissances et les régularités qu'ils ont pu observer sont des modèles appelés motifs. Les motifs s'étendent rarement sur plus d'une dizaine de positions qui traduisent généralement la présence et le voisinage proche de quelques acides aminés indispensables au fonctionnement de la protéine. Des programmes tels que Pratt (Jonassen et Higgins, 1995) ont été construits de manière à générer des motifs réguliers à partir de jeux de séquences. Cependant, les motifs ne semblent pas suffisamment expressifs au vu de la difficulté à représenter une famille et ses sous-ensembles. De plus, les motifs ne résistent pas assez au bruit lorsqu'il s'agit d'évaluer leur capacité d'apprentissage. C'est ce que nous pouvons constater de par leurs performances sur des familles de protéines dont les séquences sont peu similaires.
Cette thèse va tenir son originalité dans le fait d'introduire une nouvelle méthode de modélisation des familles de protéines. En effet, à partir d'un choix initié par B. Idmont (Idmont, 2002) et D. Fredouille (Fredouille, 2003), nous montrerons en quoi les automates finis non-déterministes (NFA) semblent un modèle particulièrement adapté à la représentation des familles de protéines. Fondés sur des principes de la théorie des langages, les automates montrent un niveau d'expressivité bien plus important que les motifs courants en bioinformatique. Cette nouvelle approche est motivée à la vue de bons résultats théoriques obtenus par des méthodes d'apprentissage par fusion d'états d'automates finis déterministes (DFA) (Oncina et Garcia, 1992). En effet, cette thèse doit permettre la présentation d'une méthode d'apprentissage automatique de notre modèle cible, et l'adaptation des méthodes connues sur DFA vers les NFA.
En outre, nous verrons que les méthodes d'apprentissage de NFA nécessitent une phase d'alignement de positions assez spécifique à notre approche. Nous constatons que les méthodes d'alignement multiple classiques mettent en avant soit des alignements complets des positions de toutes les séquences de l'échantillon d'apprentissage, soit des alignements de domaines très localisés mais partagés par l'ensemble des séquences. Nous exposerons donc un nouveau type d'alignement, nommé PLMA, correspondant à l'identification de zones significativement similaires recouvrant partiellement et localement le jeu d'apprentissage.
Il existe plusieurs applications témoignant des difficultés à obtenir un modèle expres-

sif, discret (lisible et exempt de probabilités) et permettant également de produire de bonnes prédictions. Parmi ces applications, nous avons par exemple la famille des ligands des TNF (Tumor Necrosis Factor). Cette famille, impliquée dans le développement de cancers, est très divergente dans la composition en acides aminés de ses membres. L'étude des récepteurs de ces ligands laisse à penser qu'il existerait des ligands non connus. Ces différents points en ont fait l'application principale de nos expérimentations.
Voyons à présent les différentes phases du travail abordé dans ce manuscrit divisé en trois parties de deux chapitres chacune.

Le chapitre 1 est une présentation synthétique de signatures bioinformatiques. Chacun de ces modèles est utilisé comme un moyen de représenter une ou plusieurs familles de protéines. Nous évoquons des modèles allant de simples séquences de référence peu expressives jusqu'à des motifs permettant de représenter des gaps ou encore des petits ensembles d'acides aminés. Pour chaque modèle, nous précisons également les qualités connues en terme d'apprentissage et de lisibilité notamment.

Le chapitre 2 présente les signatures de familles de protéines sous l'angle des connaissances théoriques informatiques issues de la théorie des langages. Ces connaissances sont indispensables à l'étude de modèles plus expressifs que ceux utilisés dans le chapitre 1. Nous y introduisons les définitions de langage, de séquence, de grammaire, d'automate déterministe et non-déterministe. Enfin nous exposons dans ce chapitre des méthodes d'apprentissage issues du domaine de l'inférence grammaticale.

Le chapitre 3 permet l'introduction d'une nouvelle méthode d'alignement de séquences. L'alignement multiple partiel et local (PLMA) obtenu dans un premier temps sera alors transmis à la phase d'apprentissage proprement dite.

Le chapitre 4 présente une nouvelle approche permettant de construire de façon automatique un nouveau type de modèle de représentation des familles de protéines. Cette approche est novatrice par le niveau d'expressivité qu'elle introduit en utilisant des automates non-déterministes. La phase de généralisation développée dans ce chapitre consiste à construire un automate puis à fusionner des états à l'intérieur de cet automate en utilisant un PLMA comme référence. Enfin nous abordons les difficultés dues au fait que l'on ait choisi de se confronter à des jeux de données réelles.

Le chapitre 5 est consacré à l'implémentation de notre approche, à l'utilisation d'une mesure de type MDL, ainsi qu'à une étude spécifique sur la famille des Viral Nucleic Acid Binding. Nous y présentons également les méthodes de validation classiquement utilisées dans ce type d'apprentissage.

Le chapitre 6 expose les différentes expériences menées. Nous présentons les protocoles suivis et les jeux de séquences des familles qui ont fait l'objet de nos expériences.

Si les premiers essais ont été pratiqués sur la famille des MIP, c'est sur la famille des TNF que nous avons étudié l'influence des divers paramètres de notre approche.

Première partie

Etat de l'Art

Chapitre 1

Familles de Protéines

Ce chapitre a pour but de présenter les différentes briques de base constituant les protéines et les familles s'y rapportant. Nous présentons également les différents modèles de représentations de ces familles couramment utilisés en bioanalyse.

1.1 Repères biologiques

Nous allons ici introduire les différents éléments biologiques impliqués dans la constitution des familles de protéines.

Chaque protéine est une macromolécule produite par un organisme vivant. Les protéines sont formées de chaînes d'acides aminés liés par des liaisons peptidiques. Elles sont généralement représentées sous forme de séquences (définition 1.1). Elles se replient en structures tridimensionnelles plus ou moins stables. Les protéines ont des tailles de plusieurs centaines d'acides aminés. Plus spécifiquement, les petites chaînes sont appelées peptides, les protéines étant des polypeptides pouvant être réunies par des ponts disulfures.

Définition 1.1 *Une séquence protéique s de longueur $|p|$ est une suite d'acides aminés telle que $s = a_1...a_x...a_{|}s|$.*

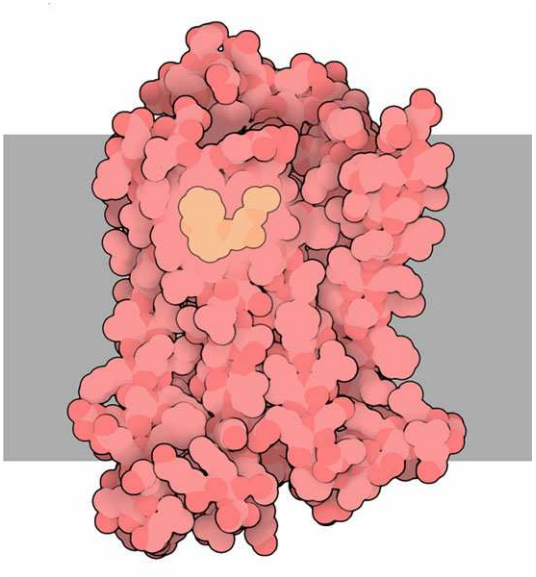


FIG. 1.1 – "Adrenergic Receptors, Molecule of the month", schéma 3D d'une protéine de la PDB (Protein Data Bank) au mois d'avril (Berman et coll., 2008).

Un acide aminé (définition 1.2) est une petite molécule élémentaire des protéines. Il en existe 20 formes différentes synthétisées par voie ribosomale dans le monde du vivant (voir 1.2). En se basant sur la composition chimique, on peut regrouper les acides aminés en 8 familles (Delepine, 2008) :

- les acides aminés aliphatiques dont le radical est une chaîne hydrogéno carbonée apolaire.
- les acides aminés hydroxylés qui portent un groupe alcool. Ils sont polaires, mais non chargés et neutres.
- les acides iminés représentés uniquement par la proline. La chaîne latérale est repliée et établit une liaison covalente avec l'atome d'azote du groupement amine.
- les acides aminés soufrés qui comportent un atome de soufre dans la chaîne latérale. L'un d'eux, la cystéine est un thiol, deux molécules de cystéine peuvent établir une liaison covalente entre leurs atomes de soufre et établir une liaison supplémentaire dans la chaîne protéique.
- les acides dicarboxyliques portent un groupement acide organique à l'extrémité de leur chaîne latérale. Ils sont donc polaires, chargés négativement (à pH neutre) et également acides.
- les acides amidés, il s'agit des versions amidées des acides aminés du groupe précédent. Le groupement OH de l'acide carboxylique est remplacé par un groupement NH2.
- les acides diaminés. La chaîne latérale porte un groupement aminé. Ils sont donc polaires, chargés positivement (à pH neutre) et basiques.
- les acides aminés aromatiques comportent un cycle aromatique dans leur chaîne

latérale. Ces molécules sont non chargées et fortement apolaires.

Cependant, les propriétés structurales et physico-chimiques de chaque acide aminé sont très variées (voir 1.5).

Définition 1.2 *Un acide aminé est une molécule organique possédant un squelette carboné et portant à la fois un groupement acide organique (COOH) et un groupement amine (NH2) basique.*

Nom	Code à 1 lettre	Code à 3 lettres	Masse molaire ($g.mol^{-1}$)	pI	pK_{a_1} (-COOH)	pK_{a_2} ($-NH_2$)	pK_{a_R} (-R)	Abondance relative[1]
Alanine	A	Ala	89.09	6.11	2.35	9.87		7.86
Arginine	R	Arg	174.20	10.76	1.82	8.99	12.48	5.39
Asparagine	N	Asn	132.12	5.41	2.14	8.72		4.15
Aspartate	D	Asp	133.10	2.85	1.99	9.90	3.90	5.34
Cystéine	C	Cys	121.16	5.05	1.92	10.70	8.18	1.51
Glutamate	E	Glu	147.13	3.15	2.10	9.47	4.07	6.66
Glutamine	Q	Gln	146.15	5.65	2.17	9.13		3.95
Glycine	G	Gly	75.07	6.06	2.35	9.78		6.94
Histidine	H	His	155.16	7.60	1.80	9.33	6.04	2.29
Isoleucine	I	Ile	131.17	6.05	2.32	9.76		5.91
Leucine	L	Leu	131.17	6.01	2.33	9.74		9.64
Lysine	K	Lys	146.19	9.60	2.16	9.06	10.54	5.93
Méthionine	M	Met	149.21	5.74	2.13	9.28		2.37
Phénylalanine	F	Phe	165.19	5.49	2.20	9.31		3.97
Proline	P	Pro	115.13	6.30	1.95	10.64		4.81
Pyrrolysine	O	Pyl						
Sélénocystéine	U	Sec	168.053					
Sérine	S	Ser	105.09	5.68	2.19	9.21		6.83
Thréonine	T	Thr	119.12	5.60	2.09	9.10		5.41
Tryptophane	W	Trp	204.23	5.89	2.46	9.41		1.14
Tyrosine	Y	Tyr	181.19	5.64	2.20	9.21	10.46	3.04
Valine	V	Val	117.15	6.00	2.39	9.74		6.73

FIG. 1.2 – tableau récapitulatif des 20 principaux acides aminés représentés dans le code génétique (O et U étant rares), source : wikipedia

La machinerie cellulaire produit des protéines en assemblant des acides aminés à partir d'un code génétique situé dans les chromosomes. Les chromosomes sont des brins d'ADN (définition 1.3) dont les bases élémentaires sont des nucléotides (définition 1.4). L'ADN est une longue molécule formée de deux brins complémentaires dont la structure forme une double hélice qui lui permet de se dupliquer et de se répliquer lors de divisions cellulaires.

Définition 1.3 *On représente un brin d'ADN (Acide DésoxyriboNucléqiue) b comme une séquence de taille $|b|$ de nucléotides telle que $b = n_1, ..., n_x, ..., n_{|}b|$ dont les bases nucléiques sont l'adénine, la cytosine, la guanine ou la thymine.*

Définition 1.4 *Un nucléotide est formé d'une base nucléique pouvant être l'adénine, la cytosine, la guanine, la thymine ou l'uracile. Les abréviations sont respectivement A,C,G,T, et U.*

Chaque protéine est générée à partir d'un gène (définition 1.5) qui va passer par une première étape de transcription de l'ADN en ARN (définition 1.6), puis par une étape de traduction de cet ARN selon le code de la figure 1.3 à partir de codons de taille 3.

Définition 1.5 *Un gène est une sous-séquence de nucléotides qui représentent un code de traduction vers une séquence d'ARN. Ce fragment de chromosome est situé à une position nommée locus.*

Définition 1.6 *Nous définissons un brin d'ARN (Acide RiboNucléqiue) b comme une séquence de taille $|b|$ de nucléotides telle que $b = n_1, ..., n_x, ..., n_{|b|}$ dont les bases nucléiques sont l'adénine, la cytosine, la guanine ou l'uracile.*

UUU, UUC → Phe	UCU, UCC → Ser	UAU, UAC → Tyr	UGU, UGC → Cys
UUA, UUG → Leu	UCA, UCG → Ser	UAA, UAG → Stop	UGA → Stop; UGG → Trp
CUU, CUC → Leu	CCU, CCC → Pro	CAU, CAC → His	CGU, CGC → Arg
CUA, CUG → Leu	CCA, CCG → Pro	CAA, CAG → Gln	CGA, CGG → Arg
AUU, AUC, AUA → Ile	ACU, ACC → Thr	AAU, AAC → Asn	AGU, AGC → Ser
AUG → Met	ACA, ACG → Thr	AAA, AAG → Lys	AGA, AGG → Arg
GUU, GUC → Val	GCU, GCC → Ala	GAU, GAC → Asp	GGU, GGC → Gly
GUA, GUG → Val	GCA, GCG → Ala	GAA, GAG → Glu	GGA, GGG → Gly

FIG. 1.3 – Code de traduction génétique (Rossier, 2008)

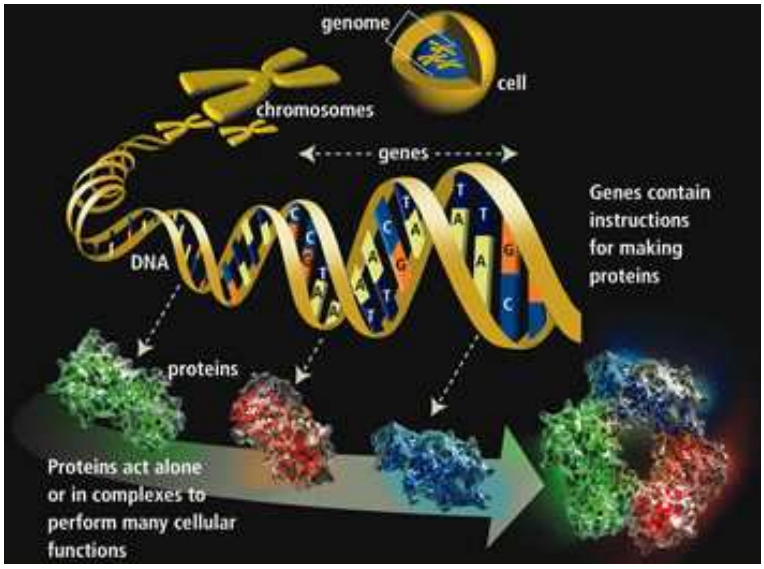


FIG. 1.4 – From gene to protein (source : US department of energy)

Les protéines interviennent à tous les niveaux dans la cellule ou les organismes vivants. On distingue plusieurs types de protéines en fonction de leur activité :

- Les enzymes sont des protéines capables de catalyser une réaction chimique précise, c'est grâce aux enzymes que des réactions nécessaires à la vie peuvent se produire à une température compatible avec l'intégrité cellulaire. On trouve notamment les protéines du métabolisme.
- Les protéines de structure construisent la charpente de l'organisme vivant. Cette famille comprend le collagène et l'elastine des animaux, la lignine des arbres, l'actine et la tubuline des muscles, etc.
- Les protéines de reconnaissance dont la fonction est de reconnaître une molécule unique et de la fixer. C'est le cas des anticorps responsables de l'élimination des antigènes, ou des protéine d'adhérence qui fixent les cellules les unes aux autres et assurent la cohérence de l'organisme.
- Les récepteurs sont capables de reconnaître une molécule dans l'environnement cellulaire et de transmettre un signal à la cellule. Les récepteurs hormonaux appartiennent à ce type.
- Les canaux sont capables de moduler la perméabilité des membranes biologiques et de trier les molécules et les ions qui traversent la membranes.

Souvent, les protéines peuvent porter plusieurs de ces propriétés simultanément et être sensibles à diverses molécules hormonales ou non, mais aussi à la tension électrique, à la pression, à la lumière, etc.

Des divers processus d'héritage et de recopie des gènes codant pour une protéine (ou ses variantes) sur un même chromosome (ou sur plusieurs chromosomes), nous pouvons

distinguer deux types de divergence que sont les gènes paralogues (définition 1.7) et les gènes orthologues (définition 1.8).

Définition 1.7 *Un gène paralogue est un gène qui au cours de son histoire subit un événement de duplication, c'est-à-dire que le même gène va être reproduit en plusieurs copies à l'intérieur d'un génome particulier.*

Définition 1.8 *Un gène orthologue est un gène qui n'est pas issu d'une duplication et qui subit uniquement des événements de spéciation au cours de l'évolution.*

On remarque que les protéines possédant des propriétés fonctionnelles communes ont souvent des structures proches. Cette similarité est également présente lorsque l'on compare les séquences de certaines de ces protéines. La raison est principalement due à l'existence d'une protéine ancestrale commune ayant possédé pour la première fois une fonction spécifique.
Considérons à présent une définition simple des familles de protéines (définition 1.9) qui regroupent sous un même nom plusieurs protéines étiquetées comme membres de cette famille. Notons que le nom de la famille est régulièrement amené à évoluer selon les différentes nomenclatures qui suivent les évolutions de la recherche en biologie.

Définition 1.9 *Une famille de protéine est un ensemble* $E = \{p_1, ..., p_x\}$ *tel que* $p_1, ..., p_x$ *sont des protéines.*

Les familles de protéines sont déclinables selon des familles fonctionnelles (de fonction identique dans la cellule), des familles structurales (de même repliement) ou encore de super-familles (regroupement de sous-familles vis-à-vis d'une propriété générale).
L'enjeu de la suite de ce chapitre est de montrer les différents modèles utilisés pour représenter les familles de protéines.

1.2 Les modèles

Sans entrer dans les détails de la théorie des langages (voir chapitre 2), nous allons cependant présenter ici différents moyens de représentation les plus répandus en bioinformatique pour exprimer des familles de protéines. A chacune de ces représentations nous associerons des règles permettant de déterminer si une séquence de protéine est acceptée ou non. Le type de représentation d'une part et les règles d'acceptations d'autre part forment ce que l'on nomme ici un modèle. Pour chacun des différents modèles qui vont être exposés, nous allons évoquer leurs atouts et leurs faiblesses en fonction des critères suivants :

- Lisibilité : le contenu du modèle est-il lisible, contenant une riche sémantique, aisément assimilable par un humain ?
- Discrimination : quelles sont les qualités du modèle en terme de capacité à accepter les membres de la famille et à rejeter les autres protéines ?

- Expressivité : le modèle est-il capable de représenter en peu de symboles des structures grammaticales complexes, partagées par de nombreux membres de la famille ?
- Apprentissage : est-ce qu'il existe des méthodes efficaces permettant de construire un modèle à partir d'échantillons de la famille ?
- Recherche : peut-on facilement utiliser le modèle dans le but de trouver de nouveaux membres dans de grandes banques de séquences ?

1.2.1 Composition en acides aminés

On utilise parfois la notion de composition en bases (nucléotides ou acides aminés) d'une séquence (définition 1.10) pour caractériser rapidement et simplement des séquences biologiques intéressantes.

Définition 1.10 *La composition d'une séquence biologique en différents éléments constitutifs de cette séquence ou selon une propriété donnée correspond à la fréquence d'apparition de cet élément ou de cette propriété.*

Par exemple, au niveau nucléotidique nous savons que les zones codantes sont souvent plus riches en G et C que la distribution observée sur le reste des brins d'ADN. Cette représentation associée à une fréquence seuil est une signature très primaire mais d'une certaine efficacité pour rechercher et localiser des régions portant des gènes dans les chromosomes.
Sur la même idée, il n'est pas rare de relever des familles de protéines que l'on dit riches en cystéines. Cet acide aminé est capable de participer à des liaisons sulfuriques avec d'autres cystéines et est souvent responsable de repliements structuraux particuliers des protéines. D'autres acides aminés sont également capables de créer un type de liaison dit peptidique entre des groupements aminés et des groupements acides.

Nous savons également que chaque acide aminé possède des propriétés physico-chimiques particulières. Ces propriétés sont rassemblées dans le diagramme de Venn de la figure 1.5. Il est alors possible d'utiliser des courbes d'hydrophobicité d'une protéine en examinant le potentiel hydrophobe ou hydrophile de chaque acide aminé positionné dans une séquence de protéine. Ces courbes forment des signatures caractéristiques de certaines familles dont on sait que les séquences qui sont amenées à être implantées dans la membrane cellulaire possèdent des régions hydrophobes au centre de la membrane et hydrophiles de part et d'autre de la membrane.
Ces modèles très simples et peu expressifs sont l'occasion de se questionner sur les similarités physico-chimiques existant entre certains acides aminés. En effet, il existe de nombreuses matrices de taille 20x20 (http ://www.genome.jp/aaindex) reflétant les fréquences de substitutions observées dans des blocs de séquences similaires, la plus célèbre étant la matrice Blosum62 (Henikoff et Henikoff, 1992). Il en existe aussi qui se basent sur des distances structurales. Enfin, les plus novatrices nous semblent être des matrices basées sur des techniques de régression à partir du recueil de nombreuses caractéristiques (Venkatarajan et Braun, 2001) ou encore des matrices basées sur des

calculs de distances analogiques (Bayoudh, 2007) entre paires d'acides aminés.

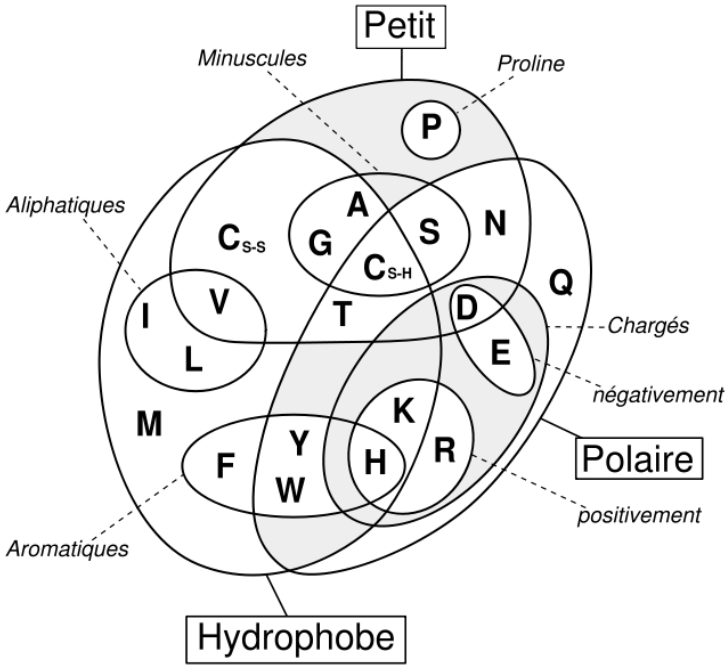


FIG. 1.5 – Diagramme de Venn des propriétés physico-chimiques des acides aminés (source : wikipedia)

En outre, les modèles les plus évolués prenant en compte différentes caractéristiques des protéines et de leurs éléments représentés sous forme de vecteurs sont les SVM (machines à vecteur support (Guermeur, 2007)).

1.2.2 Alignement de séquences

Les familles de protéines possèdent souvent quelques séquences particulièrement bien étudiées et ayant fait l'objet de plusieurs expériences en rapport direct avec la fonction connue de la famille. Ces membres ont souvent été séquencés en premier et ont une taxonomie peu exotique (homo sapiens, rattus norvegicus ou E. Coli par exemple). Si l'on considère ces séquences comme de bons représentants de la famille, une idée simple consiste à étiqueter de nouvelles séquences comme étant membres de la famille en se rapportant à un seuil de similarité.
D'où la nécessité de posséder des méthodes capables de considérer un alignement optimal entre deux séquences comme une représentation de positions conservées entre membres de la famille. Pour créer l'alignement et évaluer le degré de similarité plusieurs méthodes sont apparues suite aux premiers travaux de Smith et Watermann (Smith et Waterman, 1981). Citons Blast (Altschul et coll., 1990), et d'autres versions

plus rapides et plus précises comme Wu-blast (Gish, 1996) et Scanps (Barton, 2002). Le principe de l'algorithme principal étant d'utiliser une matrice de substitution d'acides aminés (voir 1.2.1) afin de donner un score pour chaque alignement entre deux positions de chaque séquence des deux protéines, et de maximiser la somme totale de ces scores de substitution.
Sur la figure 1.6 nous présentons une partie d'un alignement obtenu à partir du programme Blast dans sa version Blastp.

```
 Score = 89.0 bits (219),  Expect = 2e-16, Method: Compositional matrix adjust.
 Identities = 68/204 (33%), Positives = 96/204 (47%), Gaps = 19/204 (9%)

Query  12   VCGTTLHLLLLGLLLVLLPGAQGLP-GVGLTPSAAQTARQHPKMHLAHSTLKPAAHLIGD  70
            V G T    LL   ++     +  P G+ L  S AQT          +S+ KP AH++ +
Sbjct  41   VAGATTLFCLLNFGVIGPNKEEKPPNGLPLISSMAQTLTLRSSSQ--NSSDKPVAHVVAN  98

Query  71   PSKQNSLLWRANTDRAFLQDGFSLSNNSLLVPTSGIYFVYSQVVFSGKAYSPKATSSPLY  130
               +  L W +    A L +G  L +N L+VP  G+Y +YSQV+F G+          P Y
Sbjct  99   HQAEEQLEWLSQRANALLANGMDLKDNQLVVPADGLYLIYSQVLFKGQG-------CPDY  151

Query  131  --LAHEVQLPSSQYPFHVPLLSSQKMVYP-----GLQ-EPWLHSMYHGAAFQLTQGDQLS  182
              L H V  P+  Y   V LLS+ K   P     G + +PW   MY G  FQL +GD LS
Sbjct  152  VLLTHTVSRFAISYQEKVSLLSAIKSPCPKDTPEGAELKPWYEPMYLGGVFQLEKGDLLS  211

Query  183  THTDGIPHLVLSPS-TVFFGAFAL  205
               +   +L ++ S  V+FG  AL
Sbjct  212  AEVNLPKYLDITESGQVYFGVIAL  235
```

FIG. 1.6 – Alignement entre deux séquences de la famille des TNF par utilisation du programme Blast dans sa version Blastp avec les paramètres par défaut

1.2.3 Alignements multiples

A partir du moment où l'on a pu réunir suffisamment de membres à l'intérieur d'une même famille, de nouveaux modèles ont fait leur apparition. En effet, il s'agit des différentes formes de modèles basées sur des représentations par alignements multiples. Les formes les plus classiques de ces représentations sont des constructions d'alignements multiples globaux. Il s'agit d'associer sur une même colonne une position de chacune des séquences de la famille en ayant recours, lorsque c'est nécessaire, à des insertions ou des délétions. Tout comme pour les alignements entre deux séquences, il s'agit de maximiser un score global basé également sur des matrices de substitution. Nous présentons sur la figure 1.7 représentant une partie d'un alignement produit sur un échantillon de séquences provenant de la famille des TNF. Cet alignement multiple est produit par un des programmes les plus connus en alignement multiple qui se nomme ClustalW (Thompson et coll., 1994).

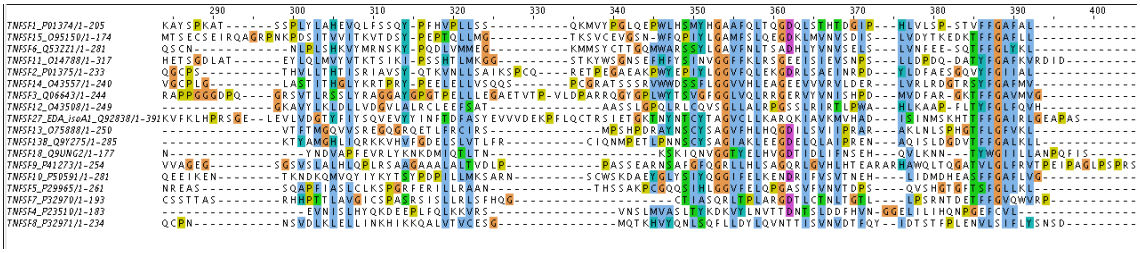


FIG. 1.7 – Alignement multiple produit par ClustalW et observé à l'aide de Jalview

Visuellement on peut observer des mutations ponctuelles intéressantes ou des zones particulièrement bien conservées, mais la somme d'informations devient rapidement un problème pour des jeux importants sur des séquences assez longues. Parmi les programmes ayant montré de bonnes performances en temps de calcul pour produire de tels alignements, nous pouvons citer Muscle (Edgar, 2004) ou encore Probcons (Do et coll., 2005).
Cependant la représentation de familles par alignement total des positions semble montrer ses limites lorsqu'elle est appliquée à des familles très divergentes en similarité. En effet, de nombreuses positions sont associées pour satisfaire l'obtention d'un score optimal d'alignement alors qu'aucune évidence de similarité, de structure ou d'implication dans une fonction particulière ne permet de valider ce rapprochement. C'est pourquoi d'autres approches d'alignement multiple se sont concentré sur des aspects purement locaux. Ainsi des programmes tels que BlockMaker (Henikoff et coll., 1995) permettent de représenter des sites actifs par l'alignement de fragments de séquences de haute similarité au sein de la famille étudiée.
Pourtant nous savons, et nous l'étudierons plus en détails dans les chapitres suivants, que le fait de créer des modèles de type automate nécessite une troisième vision des alignements entre plusieurs séquences. Nous rechercherons des alignements à la fois locaux (sur une vingtaine de positions par exemple) et partiels (ne prenant en compte que quelques séquences par familles par exemple). Cette nouvelle forme de représentation nous a été très largement inspirée par le programme d'alignement multiple Dialign2 (Morgenstern, 1999) sur lequel nous reviendrons également à plusieurs reprises.
Afin que ces représentations par alignements multiples deviennent des modèles, il a fallu leur associer des méthodes, souvent statistiques, afin de permettre de donner un score à une séquence par rapport à la représentation, ainsi qu'un seuil permettant de décider du fait que cette séquence soit membre ou non de la famille étudiée.

1.2.4 Matrices PSSM

Si l'on choisit des représentations locales produites par BlockMaker, il est possible de relever le nombre d'occurrences de chaque acide aminé pour chaque colonne du Block. Cette opération se traduit par la construction de matrices PSSM (Position specific scoring matrix) (Ben-Gal et coll., 2005). De même que pour les alignements Blast, les alignements de séquences sur des PSSM produisent des scores indiquant leur degré

de similarité avec les séquences ayant produit les PSSM.
En ce qui concerne les alignements globaux, il existe un outil particulièrement pratique qui se nomme psi-blast (Schaffer et coll., 2001). Cette version procède par un premier Blast et produit une PSSM à partir des séquences renvoyées par chaque itération et permet de rescanner à nouveau les séquences d'une banque.
Les profils de la banque de données Prosite (Hulo et coll., 2004) peuvent également être considérés comme des PSSM produites généralement sur des sites actifs.
Cependant les PSSM ne sont pas ou très peu lisibles et compréhensibles directement (les solutions les plus pratiques étant de les observer sous forme de Sequence Logo (Schneider et Stephens, 1990)). Elles sont surtout prisées pour leur capacité à reconnaître de nouveaux membres potentiels pour les familles de protéines.

1.2.5 Profils HMM

Les PSSM étant plutôt orientées vers des alignements locaux, ce sont d'autres modèles qui permettent d'utiliser les alignements multiples globaux dans le but de scanner des banques de séquences.
Ces modèles sont les profils HMM ou Hidden Markov Models. Ils ont la particularité d'être suffisamment robustes aux problématiques de mutations (insertion, délétion en particulier) y compris sur de grandes portions de séquences. Les profils HMM sont basés sur un squelette simple d'automate probabiliste (voir figure 1.8). Ainsi des approches comme Hmmer (Eddy, 1998) permettent des apprentissages très sensibles et très compétitifs pour ce qui est de la reconnaissance et de la discrimination entre différentes familles. Il existe des bases de données mettant à disposition des profils HMM entraînés sur des familles. Pfam (Bateman et coll., 2004) est l'une de ces bases. Cependant, là aussi toutes les séquences sont mises en correspondance pour faire émerger les zones les plus similaires et les plus partagées par les séquences de la famille. Ces modèles statistiques sont très efficaces en reconnaissance des membres d'une famille même lorsqu'il existe des divergences de similarité entre les séquences membres. Cependant l'apprentissage est effectué sur une représentation ne permettant pas de visualiser les zones importantes marquant l'apparition de sous-familles. De plus, la difficile lecture de modèles statistiques comme les PSSM est ici beaucoup plus forte car les profils HMM sont de véritables boîtes noires n'apportant pas d'informations sémantiques. De plus ce qu'un HMM permet potentiellement en expressivité n'est pas utilisé par ces profils HMM.

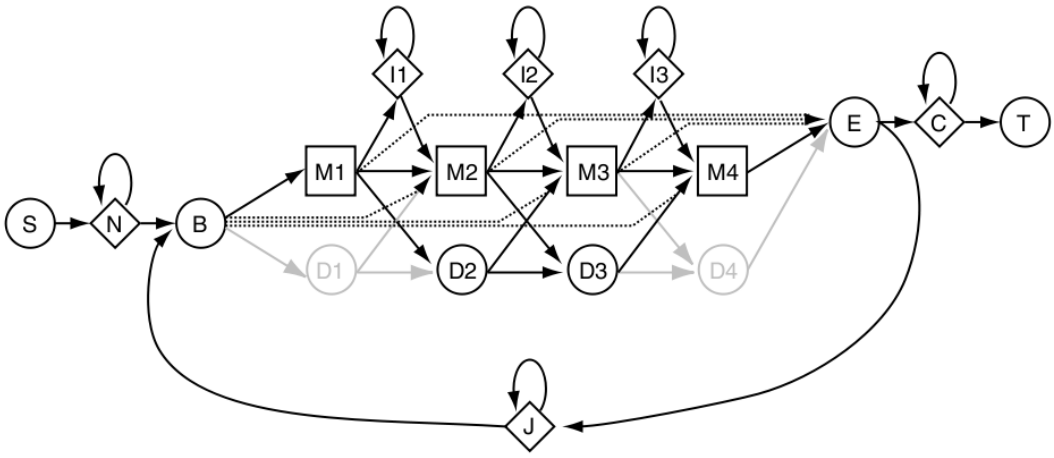


FIG. 1.8 – The Plan7 architecture (HMM architecture (Eddy, 1998). Les carrés indiquent des états de match. Les losanges indiquent des états d'insertion. Les cercles indiquent les états de délétion.

1.3 Les Motifs courts

A l'oppposé de ces modèles statistiques, il existe des représentations dites par motifs. Ces motifs sont des sous-expressions régulières permettant de décrire, par un codage particulier, une suite d'acides aminés avec, selon le niveau d'expressivité (voir figure 1.9 tirée de (Jonassen et Higgins, 1995)), la possibilité de représenter des répétitions, des gaps ou encore des groupes d'acides aminés. Toutes ces informations permettant de créer une signature caractéristique de quelques positions conservées chez tous les membres d'une même famille.

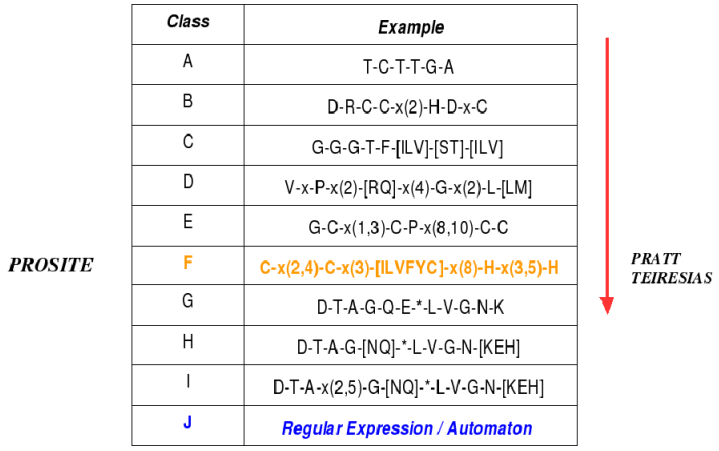


FIG. 1.9 – Le symbole "x" représente un acide aminé quelconque ; "x(2)" représente exactement 2 acides aminés ; "x(1,3)" représente entre 1 et 3 acides aminés ; " * " représente de 0 à une infinité d'acides aminés ; "[ILV]" représente le choix entre les acides aminés I, L ou V.

La banque de données Prosite (Hulo et coll., 2004) est une source de référence en matière de motifs de familles de protéines. Ces motifs ont été obtenus de façon semi-automatique par alignement multiple et analyse humaine.
Cependant il existe des méthodes de découverte automatique de ce type de motifs. L'une des plus connues est sûrement celle de PRATT (Jonassen et Higgins, 1995). PRATT est un programme qui a été développé par l'équipe d'I.Jonassen à l'Université de Bergen. Il renvoie des motifs dont le pouvoir d'expression se situe au niveau de la classe F, ce qui lui permet de donner des représentations fines des motifs. Son algorithme se base sur la construction d'un arbre dont les noeuds sont les motifs potentiels. L'arbre possède une racine vide et s'étend par un ajout récursif d'acides aminés à chaque étage par une approche Bottom-up (Brazma et coll., 1995). Un score est calculé en fonction de la qualité du motif et en particulier du nombre de séquences sur lesquelles il est validé. Ce raffinement par heuristique (Jonassen et coll., 1996) permet une grande vitesse d'exécution, sachant que le choix du chemin est effectué a priori sur ce qui semble le plus prometteur. Cependant, il faut noter qu'en pratique le backtracking n'est pas forcément effectué et donc que l'algorithme n'explore pas toutes les solutions. Dans PRATT, d'autres biais comme les matrices de substitution (la possibilité d'introduire les fréquences de substitution entre acides aminés) peuvent encore améliorer l'heuristique. L'utilisateur garde le contrôle du paramètre principal en déterminant le nombre de séquences minimum validant le motif. On peut aussi influer sur le niveau d'indéterminisation de la découverte, c'est-à-dire le seuil de mutations que l'on s'autorise à observer.
L'algorithme de PRATT extrait des motifs possédant des gaps puis il effectue une étape de raffinement. Par un parcours de l'ensemble des séquences, il peut éventuelle-

ment remplacer ce qui était considéré comme indéterminé par des acides aminés comme "[LIVM]" partageant cette même position.
La version 2 du programme PRATT est une amélioration qui permet entre autre de lancer une exécution à partir d'alignements multiples. Il est aussi possible de choisir une séquence de référence qui doit posséder au moins une occurrence d'un motif avant que celui-ci ne soit validé. Mais il est surtout plus efficace en raison d'heuristiques et en fonction de scores mieux adaptés.

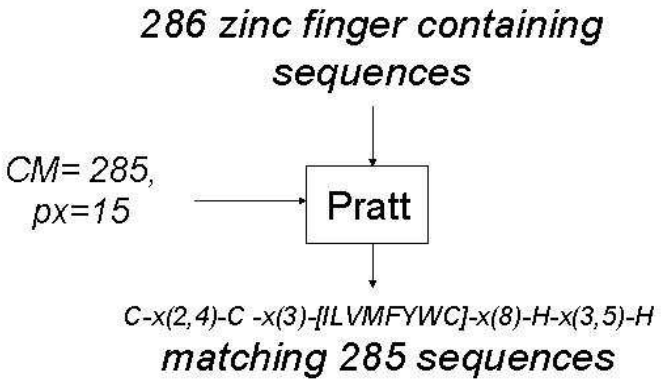


FIG. 1.10 – Exemple de PRATT sur la famille des Zinc Finger

D'autres programmes de ce type existent. Certains comme Teireisias sont directement dérivés de méthodes linguistiques.
Enfin, nous pouvons également citer le programme Stan (Nicolas et coll., 2005) développé dans l'équipe Symbiose. Il permet de chercher en une expression deux motifs courts à plusieurs nucléotides d'intervalles afin de scanner potentiellement plusieurs phases à la fois.

1.4 Les combinaisons de modèles

Les modèles les plus récents permettent des combinaisons de certains modèles exposés précédemment. Citons la variante Phi-Blast du programme Blast, qui permet d'associer à la fois un alignement d'une séquence, et l'utilisation d'un motif filtre.
Nous avons également pu découvrir que la base Pfam possédait des entrées nommées des *clans* (Finn et coll., 2006). Un clan contient deux familles ou plus de Pfam. Cette composition résulte de nombreuses séquences membres de plusieurs familles. Nous voyons là une limite des profils HMM qui sont par définition des profils consensus de l'ensemble de l'échantillon et donc incapable de présenter correctement différentes sous-familles dans un même modèle.

Chapitre 2

Approche de la théorie des langages

Nous avons vu, dans le chapitre 1, différents modèles de familles de protéines. Ces modèles étant composés de représentations des familles et de règles permettant de juger de l'appartenance d'une séquence à ces familles.
Afin de formaliser ces notions, nous présentons dans ce chapitre le domaine de la théorie des langages. Cette théorie établit différents niveaux de complexité de représentations des langages. Pour chaque niveau il existe des catégories de machines ou de grammaires associées. Les représentations présentées précédemment étaient pour la plupart de niveau assez faible en expressivité et ne nécessitaient pas forcément une introduction à la théorie des langages. Cependant le modèle étudié dans cette thèse est de type automate, soit un niveau d'expressivité plus important. C'est pourquoi nous introduirons des notions de théorie des langages. Puis nous présenterons les automates à la lumière de cette théorie. Nous étudierons des méthodes permettant l'apprentissage automatique d'automates. Et enfin nous verrons les adaptations permettant une application à la biologie et aux séquences de protéines.

2.1 Théorie des langages

On peut définir un langage comme un ensemble de mots sur un alphabet donné. Nous introduisons ici les définitions précises de ces concepts et les notations utilisées au travers des différentes représentations de la théorie des langages.

Définition 2.1 (Alphabet) *Un alphabet est un ensemble fini non-vide de symboles. Soit Σ cet ensemble, la taille de Σ, notée $|\Sigma|$ est égale au nombre de symboles de l'ensemble.*

Définition 2.2 (Mot) *Un mot w sur un alphabet Σ est une suite $x_1...x_n$ finie de lettres x_i de Σ. Un mot est aussi nommé séquence ou chaîne. Sa taille, notée $|w|$, est la longueur de la séquence en nombre de symboles.*

Définition 2.3 (Langage) *Un langage est un ensemble de mots.*

Ainsi lorsque l'on travaille sur des ensembles de séquences, nous travaillons sur des ensembles nommés langages dans le cadre de la théorie des langages. La théorie des langages s'est particulièrement intéressée aux modèles permettant de représenter et définir un langage (dit alors formel). Nous introduisons ici les représentations par grammaires selon la hiérarchie de Chomsky (2.1.1), les expressions régulières (2.1.3) et les automates (2.1.2).

2.1.1 Hiérarchie de Chomsky

Les grammaires formelles sont des systèmes de réécriture permettant de représenter des langages potentiellement infinis.
Les éléments constitutifs de ces grammaires sont : un ensemble de symboles terminaux noté T, un ensemble de symboles non-terminaux noté N, un ensemble de règles noté R et enfin l'axiome (symbole non terminal de départ noté S).
Dans la suite on notera $V = N \cup T$, et on supposera que N et T sont disjoints.
Selon Chomsky (Chomsky, 1956), les grammaires sont constituées en cinq classes (type 0 à type 4) hiérarchiquement imbriquées. Les grammaires de type 0 sont les plus générales et incluent donc les grammaires de type 1 (dites "contextuelles") qui incluent elles-mêmes les grammaires de type 2 (dites "hors-contexte") qui incluent eux-mêmes les grammaires de type 3 (dites "régulières").

Les grammaires générales (définition 2.4) représentent des langages récursivement énumérables, mais le temps pour savoir si une phrase appartient ou non à celle-ci n'est pas forcément fini. Elles sont reconnaissables par machine de Turing.

Définition 2.4 *Les grammaires générales (type 0)*
sont de la forme $w1 \mapsto w2$ *où* $w1$ *et* $w2 \in (N \cup T)*$
$\alpha \rightarrow \beta$
$\alpha, \beta \in V^*, \alpha \neq \epsilon$

Les grammaires contextuelles (définition 2.5) sont appelées ainsi car le remplacement d'un élément non-terminal peut dépendre des éléments autour de lui : son contexte. Les langages produits, appelés langages contextuels, sont exactement ceux reconnus par machine de Turing linéairement bornée non déterministe.

Définition 2.5 *Les grammaires contextuelles (type 1)*
sont de la forme : $sXt \mapsto swt$ *où* s, w *et* $t \in (N \cup T)*$, $X \in N$ *et* w *n'est pas vide :* $w \in (N \cup T)+$.
$\alpha A \beta \rightarrow \alpha \gamma \beta$
$A \in N, \alpha, \beta, \gamma \in V^*, \gamma \neq \epsilon$

Les grammaires hors-contexte (définition 2.6) signifient que les éléments non-terminaux sont traités individuellement (sans dépendance contextuelle). Ces grammaires produisent exactement les langages hors-contexte, reconnaissables par automate à pile. En France on parle aussi de langages algébriques.

Définition 2.6 *Les grammaires hors-contexte (type 2)*
sont de la forme : $X \mapsto w$ *où* $X \in N$ *et* $w \in (N \cup T)*$
$A \rightarrow \gamma$
$A \in N, \gamma \in V^*$

L'ensemble formé par les grammaires linéaires gauche et droite est l'ensemble des grammaires régulières (définition 2.7). Les langages rationnels sont exactement ceux reconnus par automate fini.

Définition 2.7 *Les grammaires régulières rationnelles (type 3)*
Il faut distinguer deux cas (qui sont équivalents).
** Les grammaires linéaires gauches. Les règles sont de la forme :*
$A \rightarrow Ba$
$A \rightarrow a$
$A, B \in N, a \in \Sigma$
** Les grammaires linéaires droites. Les règles sont de la forme :*
$A \rightarrow aB$
$A \rightarrow a$
$A, B \in N, a \in \Sigma$

Il existe enfin une classe très restreinte dite grammaire à choix finis (définition 2.8), qui est souvent omise dans la hiérarchie.

Définition 2.8 *Les grammaires à choix finis (type 4)*
sont de la forme : $X \mapsto a$ *où* $X \in N$ *et* $a \in T$

Cette thèse concerne principalement un niveau de langage et de grammaire de type 3. Les grammaires rationnelles sont une forme des représentations des langages réguliers. Cependant il existe des représentations de puissance équivalente à ce niveau d'expressivité. Nous allons voir en premier un type de machine classiquement utilisé et également équivalent : les automates d'états finis. Nous verrons également une forme de représentation par expressions régulières dont les motifs forment un sous-ensemble.

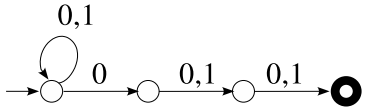
2.1.2 Automates à états finis

Les automates à états finis peuvent être vus comme une représentation graphique des grammaires de type 3 faisant correspondre aux non-terminaux des états et aux règles de production des transitions entre ces états.

Définition 2.9 (NFA) *Un automate fini non déterministe (NFA), est un quintuplet* $A = \langle \Sigma, Q, I, \delta, F \rangle$ *tel que :*
- Σ *est l'alphabet d'entrée*
- Q *est l'ensemble fini d'états*
- $I \subseteq Q$ *est l'ensemble des états initiaux*
- δ *est la fonction de transition de l'automate définie de* $Q \times \Sigma$ *vers* 2^Q*, un triplet* $\langle q, a, q' \rangle$ *avec* $\delta(q, a) \in Q'$ *est appelé transition*
- F *est l'ensemble des états finals*

On généralise classiquement la fonction δ *aux mots et aux ensembles d'états, soit de* $2^Q \times \Sigma^*$ *vers* 2^Q *par :* $\forall q \in Q$, $\forall w \in \Sigma^*$, $\forall a \in \Sigma$, $\delta(q, \epsilon) = q, \delta(q, aw) = \bigcup_{q' \in \delta(q,a)} \delta(q', w)$ *et* $\forall Q' \subseteq Q$, $\delta(Q', w) = \bigcup_{q \in Q'} \delta(q, w)$.

Le langage représenté par un NFA A, noté $L(A)$, est l'ensemble des mots w de Σ^* tel que $\delta(I, w) \cup F \neq \emptyset$. Un exemple de NFA est donné figure 2.1.



Un NFA pour $\Sigma = \{0, 1\}$ représentant le langage de tous les mots ayant un 0 comme antépénultième lettre ($\Sigma^* O \Sigma^2$). Les états sont représentés par des cercles, les états initiaux sont caractérisés par une flèche entrante sans symbole, les états finals ont un double cercle, les transitions sont représentées par des arcs orientés entre les états étiquetés par des lettres de l'alphabet.

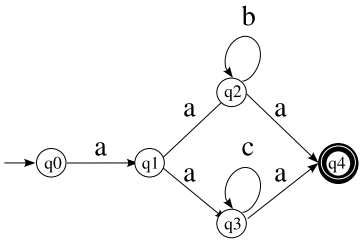
FIG. 2.1 – Exemple de NFA.

La notion d'acceptation d'un mot ou d'une séquence, dans un NFA représentant le langage, se définit formellement à partir de la notion de chemin.

Définition 2.10 (Chemin) *Un chemin pour un mot* $w = a_1 ... a_{|w|}$ *dans un automate* $A = \langle \Sigma, Q, I, \delta, F \rangle$ *entre les états* q *et* q' *est une séquence de* $|w| + 1$ *états, dénotée* $(q_0, ..., q_{|w|})_w$ *telle que* $q_0 = q, q_{|w|} = q'$ *et* $\forall i \in [0, |w| - 1], q_{i+1} \in \delta(q_i, a_{i+1})$. *On dira des transitions* $\langle q_i, a, q_{i+1} \rangle$ *qu'elles sont exercées par le chemin.*

Définition 2.11 (Acceptation) *Une acceptation pour un mot* $w = a_1 ... a_{|w|}$ *dans un automate* $A = \langle \Sigma, Q, I, \delta, F \rangle$ *est un chemin* $(q_0, ..., q_{|w|})$ *dans* A *entre les états* q_0 *et* $q_{|w|}$ *tel que* $q_0 \in I$ *et* $q_{|w|}$ *est final, i.e.* $q_{|w|} \in F$. *L'état* q_0 *est nommé état initial de l'acceptation et l'état* $q_{|w|}$ *état final de l'acceptation.*

On note $Acc_A(w)$ l'ensemble des acceptations d'un mot w dans un automate A. La notion de chemin et d'acceptation est illustrée figure 2.2.



Il y a deux chemins entre q_1 et q_4 par aa : (q_1, q_2, q_4) et (q_1, q_3, q_4) et une acceptation pour le mot $aaba$ dans l'automate : (q_1, q_2, q_2, q_4).

FIG. 2.2 – Illustration de la notion de chemin et d'acceptation.

Les automates finis déterministes (DFA, définition 2.12) sont un sous-ensemble des automates finis non déterministes. Nous verrons deux exemples d'algorithmes d'apprentissage de DFA par la suite. Cependant, dans notre approche, nous avons fait le choix de ne pas nous restreindre à ce sous-ensemble malgré une littérature plus importante sur les DFA. Nous en expliquerons les raisons.

Définition 2.12 (DFA) *Un automate fini déterministe (DFA), est un quintuplet* $A = \langle \Sigma, Q, I, \delta, F \rangle$ *tel que :*

- Σ *est l'alphabet d'entrée*
- Q *est l'ensemble fini d'états*
- $I \subseteq Q$ *est l'ensemble des états initiaux*
- δ *est la fonction de transition de l'automate définie de* $Q \times \Sigma$ *vers* Q*, un triplet* $\langle q, a, q' \rangle$ *avec* $Q' \in \delta(q, a)$ *est appelé transition*
- F *est l'ensemble des états finals*

Nous présentons également la définition 2.13 d'un autre type d'automate nommé FTA. En effet, les FTA ont été utilisés par A. Leroux dans un contexte similaire au nôtre (voir section 2.4).

Définition 2.13 (FTA) *Un automate à transitions fini non déterministe (FTA), est un triplet* $A = \langle \Sigma, Q, T$ *tel que :*

- Σ *est l'alphabet d'entrée*
- Q *est l'ensemble fini d'états*
- T *est l'ensemble des transitions, chaque transition du FTA est un quadruplet* $t = (q, l, m, q')$*, tel que* $q \in Q$ *(resp.*$q' \in Q$*) est l'origine (resp. la cible) avec* $l \in \Sigma$ *le label de la transition, et* m *la marque indiquant si* t *est un état initial et/ou final*

2.1.3 Expressions régulières

La représentation des langages sous forme d'expression régulière (ou expression rationnelle) est une notation particulière des langages réguliers. Les expressions régulières sont définies par le formalisme suivant (Yu, 1997).

Définition 2.14 (Expression régulière) *Soit Σ un alphabet. Une expression régulière e sur Σ et le langage $L(e)$ qu'elle représente sont définis récursivement comme suit :*

1. *$e = \emptyset$ est une expression régulière représentant le langage $L(e) = \emptyset$.*
2. *Pour tout $a \in \Sigma \cup \epsilon$, $e = a$ est une expression régulière représentant le langage $L(e) = a$.*
3. *Si e_1 et e_2 sont des expressions régulières représentant respectivement les langages $L(e_1)$ et $L(e_2)$, alors $(e_1 + e_2)$, $(e_1.e_2) = (e_1e_2)$, e_1^* sont des expressions régulières représentant respectivement les langages $L(e_1) \cup L(e_2)$, $L(e_1) \odot L(e_2)$, $L(e_1)^*$ (l'opération $R \odot S$ dénote le produit de Cauchy, défini par : $L \odot L' = uv \in \Sigma^* / u \in L, v \in L'$).*

Par commodité on utilisera un système de priorité sur les opérateurs (par priorité décroissante, $*$, ., $+$) et quelques abus de notation :

- L'expression régulière $e = \Sigma^*$ représente le langage $L(e) = a^* / a \in \Sigma$.
- L'expression régulière $e = \Sigma^n$ représente le langage $L(e) = w \in \Sigma^* / |w| = n$.
- L'expression régulière $e = \Sigma^{n,m}$ avec $n, m \in \mathbb{N}$ représente le langage $L(e) = w \in \Sigma^* / |w| = [n, m]$.

Par exemple, le langage, sur l'alphabet $\Sigma = a, b$, de tous les mots contenant le sous mot *baba* peut s'écrire $\Sigma^* baba \Sigma^*$, le langage contenant tous les mots avec un nombre pair de a peut s'écrire $(b^*|ab^*a)^*$.

2.2 Les séquences génomiques au travers de la théorie des langages

L'analyse de séquences en bioinformatique se rapporte principalement aux séquences d'ADN, d'ARN ou de protéines. Cette analyse se fait traditionnellement sous l'angle de comparaisons par alignement des séquences ou par découverte de motifs courts conservés (voir chapitre 1).
La théorie des langages nous permet ici de faire ces analyses sous un angle plus large. Cependant nous nous focalisons ici sur les applications des langages réguliers qui sont une première avancée, assez peu exploitée, vers plus d'expressivité que les outils standards.

2.2.1 Description des données

Les briques de base des données biologiques que nous avons à considérer dans les jeux de séquences, sont les acides nucléiques et les acides aminés. Nous avons donc à disposition trois alphabets.

- A,G,C,T : Alphabet de taille 4 des acides nucléiques de l'ADN
- A,G,C,U : Alphabet de taille 4 des acides nucléiques de l'ARN

– A,R,N,D,C,E,Q,G,H,I,L,K,M,F,P,S,T,W,Y,V : Alphabet de taille 20 des acides aminés des protéines

Les mots sont les séquences composées à partir de ces alphabets. En bioinformatique, nous parlons plus volontiers de séquences que de mots. Nous avons donc des séquences nucléiques et des séquences protéiques. Même si les séquences nucléiques sont actuellement traitables par notre approche, nous nous sommes plutôt focalisé sur les séquences protéiques. Ce type de séquences a été favorisé de part nos collaborations orientées "familles de protéines" ainsi que par la richesse des informations structurelles et physico-chimiques.

2.2.2 Le langage d'une famille de protéines

D'après la définition théorique d'un langage (2.3), on peut faire correspondre à chaque famille de protéines un langage représenté par l'ensemble S des séquences appartenant à la famille. Chaque séquence de S est associée à une structure ou une fonction symbolisant la famille.
En réalité, la nature n'a pas eu le temps et l'espace d'explorer toutes les possibilités et variantes de séquences pouvant appartenir à S. D'autre part, en pratique, les projets de séquençage n'ont pas extrait toutes les possibilités et variantes de séquences existant sur Terre à toutes les époques. C'est pourquoi, il est essentiel d'avoir à l'idée les différents ensembles de séquences :

– l'ensemble de séquences connues, séquencées et annotées par les biologistes ($L1$)
– l'ensemble de séquences connues, mais pas encore annotées ($L2$)
– l'ensemble de séquences existant dans la nature mais non-connues et non séquencées ($L3$)
– l'ensemble de séquences non-explorées par la nature mais qui, si l'on synthétisait leur séquence, seraient fonctionnelles ($L4$)

Nous verrons dans la partie correspondant à l'apprentissage du modèle que le but de notre approche va être d'apprendre le langage $L5 = L1, L2, L3, L4$ à partir de $L1$ lorsque l'on a la chance de posséder un ensemble $L1$ suffisamment précis et fiable. L'espoir étant de pouvoir rapidement annoter $L2$ et de permettre de rechercher plus facilement de nouvelle séquences appartenant à $L3$.

2.2.3 Expression régulière en bioinformatique

Les prémices, assez peu expressives, des expressions régulières ont été rapidement adaptées à la bioinformatique. Avec l'introduction de la description des gaps, ces représentations ont été appelées "motifs". Nous avons constaté que le niveau d'expressivité potentiel des expressions régulières n'est pas entièrement exploité. En effet, la célèbre banque de motifs Prosite (Hulo et coll., 2004), ainsi que l' outil de découverte de motifs Pratt (Jonassen et Higgins, 1995) représentent tous deux des langages de classe J dans la représentation de Jonassen. La classe J est moins expressive que ne le permet la classe des expressions régulières (voir figure 2.3.3 dans le chapitre précédent).

2.2.4 Automates en bioinformatique

La représentation par automate n'est pas forcément familière pour les biologistes. Pourtant les automates peuvent se révéler comme des modèles très intéressants. Basés sur l'alphabet des acides aminés, ils représentent de manière discrète et visuelle, le langage de l'ensemble des séquences protéiques que l'on souhaite considérer.

L'observation d'une séquence à l'intérieur d'un automate bien construit doit pouvoir nous renseigner sur les positions importantes partagées par la famille, ou les domaines utilisés, ou encore les zones structurales possibles. De plus les automates sont des représentations permettant de faire apparaître des sous-familles divergentes dans le même modèle.
La représentation des langages protéiques par automate présentant de bonnes propriétés, le but de cette thèse a été de faire de l'apprentissage automatique d'automates.

2.3 Apprentissage d'automates d'états finis

2.3.1 Introduction

La section précédente nous a permis de définir et d'établir la hiérarchie des grammaires, et en particulier les NFA. Nous allons à présent étudier les domaines et les méthodes permettant l'apprentissage automatique de ces grammaires.

L'apprentissage d'automates s'inscrit dans le cadre de l'inférence grammaticale qui fait elle-même partie du domaine de l'apprentissage par induction. Ce dernier peut être formulé comme la tâche de découvrir un concept caractérisant un ensemble d'objets à partir d'exemples de ces objets (et parfois de contre-exemples). L'espace des concepts est fixé et on s'intéresse en général aux concepts optimaux, selon un critère lié à la complexité de la représentation du concept.
En inférence grammaticale, les objets que l'on considère sont des séquences sur un alphabet fini. Au concept recherché, correspond alors un ensemble de séquences, et donc un langage formel. Afin de caractériser ce langage, on utilise des représentations syntaxiques comme les grammaires, d'où le terme d'inférence grammaticale.
L'inférence grammaticale est un domaine de recherche né dans les années 1960, motivé par l'analyse et l'acquisition de la langue naturelle. L'article fondateur est celui de E.M. Gold (Gold, 1967).
Dans le cas plus spécifique de l'apprentissage d'automates (représentation graphique des grammaires régulières), il existe un nombre important de publications proposant des algorithmes d'inférence : (Biermann et Feldman, 1972; Oncina et Garcia, 1992; Angluin, 1987; Yokomori, 1994; Denis et coll., 2001; Lang, 1992).
Néanmoins, parmi tous ces travaux, seuls (Yokomori, 1994; Denis et coll., 2001) utilisent la représentation par automates finis non déterministes, tous les autres se limitant à l'utilisation d'automates déterministes.
En ce qui concerne la bioinformatique, les travaux de D. Fredouille (Fredouille, 2003)

nous ont montré les premières applications d'apprentissage de NFA sur des séquences protéiques. Ils ont montré en quoi le choix des NFA était adapté au contexte étudié. Nous y revenons ci-après.

2.3.2 Le choix du non déterminisme

Les automates déterministes possèdent des propriétés facilitant leur manipulation. D'un autre côté, les arguments principaux de l'utilisation des automates non déterministes sont leur compacité et leur caractère explicite. En effet, le critère optimisé par les algorithmes d'inférence d'automates est habituellement, par application du principe du rasoir d'Occam, la taille des automates (en nombre d'états). Le nombre d'états d'un automate non déterministe nécessaire pour représenter un langage régulier peut être exponentiellement plus petit que celui de l'automate déterministe minimal pour ce langage (Hopcroft et Ullman, 1979). Les automates non déterministes ont donc un avantage certain sur les déterministes pour permettre l'inférence de représentations de petite taille.
Le deuxième avantage des automates non déterministes est de pouvoir représenter le même langage de façons très différentes. On pourra ainsi choisir, parmi les automates possibles pour un langage, celui qui est le plus facilement compréhensible, rendant ce mode de représentation explicite. C'est la capacité à conserver la sémantique qui donne un intérêt particulier aux NFA.
C'est pourquoi notre approche se basera sur l'apprentissage de NFA et nous n'aborderons pas l'apprentissage de DFA dans notre approche décrite au chapitre 3. Seul ce chapitre décrira les bases de l'apprentissage d'automates à partir d'algorithmes classiques connus sur les DFA.

2.3.3 Algorithmes par fusion d'états

L'apprentissage d'automates dans le cadre de l'inférence grammaticale consiste à trouver le langage cible L étant donné un jeu d'apprentissage $S+$ de séquences appartenant à ce langage et parfois en utilisant également un jeu de contre-exemples $S-$.
Les algorithmes classiques, dont nous verrons deux représentants RPNI et EDSM, fonctionnent par *fusion d'états* à partir d'un automate canonique maximal (MCA). Chaque séquence s_i de $S+$ est représentable sous la forme d'un automate canonique A_i décrivant un unique chemin acceptant s_i.

Définition 2.15 (Automate Canonique) *Soit une séquence s, l'automate canonique $A(s)$ est le plus petit automate déterministe complet reconnaissant le langage $\{s\}$.*

Le MCA de $S+$ est l'automate acceptant entièrement et uniquement le langage des séquences d'apprentissage $S+$. Nous définissons alors le MCA de la manière suivante.

Définition 2.16 (Automate Canonique Maximal (MCA)) *Pour $A()$ une fonction qui renvoie l'automate canonique d'une séquence, $MCA(S+) = \cup_{i=1}^{i=|S+|} A(s_i)$.*

Le MCA de $S+$ représente exactement le langage de $S+$. L'idée de l'apprentissage par fusion d'états (définition 2.17) est de généraliser le langage (augmenter le nombre de séquences acceptées) en appliquant des fusions entre états. En effet, les fusions entre les états génèrent de nouveaux automates introduisant de nouveaux chemins et donc de nouvelles séquences dans le langage, à partir du MCA.

Définition 2.17 (Fusion d'Etats) *Soit un automate A. La fusion de deux états Q_1 et Q_2 de A crée un nouvel état Q_3 tel que $\forall x \in \Sigma, \delta(Q_3, x) = \{\delta(Q_1, x) \cup \delta(Q_2, x)\}$ et $\forall i \in Q$ si $\exists \delta(Q_i, y) = Q_j$ avec $j = 1$ ou $j = 2$ alors $\delta(Q_i, y) = Q_3$. La fusion entraîne également la disparition des états fusionnés ainsi que des arcs arrivant ou sortant de ces états.*

Le cas extrême de généralisation peut être atteint lorsque tous les états ont été fusionnés. L'automate résultant s'appelle alors l'automate universel (2.18).

Définition 2.18 (Automate Universel (UA)) *L'UA est un automate défini sur un alphabet Σ, à un état et une transition bouclant sur l'unique état, qui accepte tous les mots possibles définis sur l'alphabet Σ, soit : Σ^*.*

La méthode d'inférence grammaticale basée sur la fusion d'état consiste alors à explorer l'espace de recherche (figure 2.3.3) des automates possibles entre le MCA et l'UA.
Certains algorithmes (RPNI par exemple) peuvent également prendre en compte un échantillon d'apprentissage négatif $S-$. La mise à disposition de cet échantillon d'apprentissage négatif permet de limiter l'espace de recherche. Cette frontière est nommée "border set" et est atteinte dès qu'une séquence négative est acceptée par l'automate courant dans le processus de fusion.

L'espace de recherche pour l'inférence de DFA par fusion d'états est décrit avec précision par P. Dupont, L. Miclet et E. Vidal (Dupont et coll., 1994).

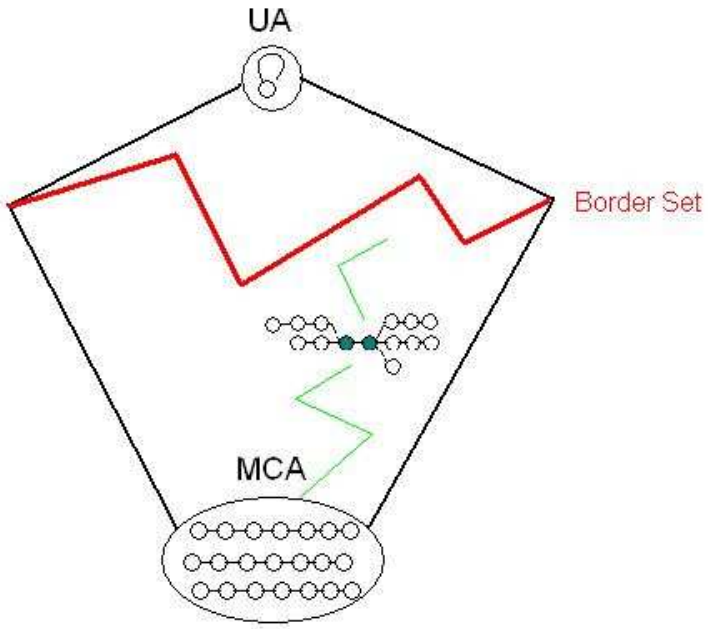


FIG. 2.3 – Espace de recherche pour un automate fini non déterministe.

Nous donnons ici le principe général des algorithmes gloutons par fusion d'états lorsque l'on dispose d'un jeu d'apprentissage positif, d'un jeu d'apprentissage négatif et d'une fonction permettant de tester la compatibilité entre un automate et le jeu d'apprentissage négatif de manière à ne passer franchir le Border Set.

Algorithme 1 Principe des algorithmes gloutons par fusion à partir d'un ensemble $S+$ et d'un ensemble de contre-exemples $S-$

Entrées jeu d'apprentissage positif $S+$, jeu d'apprentissage négatif $S-$
Sorties automate résultat A
  Tant que $choix\_de\_deux\_etats(A, Q_1, Q_2)$ **faire**
    $A' \leftarrow fusion(A, Q_1, Q_2)$
    Si $est_compatible(A', S-)$ **alors**
      $A \leftarrow A'$
    Fin si
  Fin tant que
  $Retourner(A)$.

2.3.4 Algorithmes d'inférence de DFA et de NFA

Les algorithmes d'inférence par fusion d'états les plus connus sont certainement les algorithmes RPNI (Oncina et Garcia, 1992) et EDSM (Lang et coll., 1998b). Ces deux algorithmes infèrent des DFA.
RPNI et EDSM fonctionnent tous deux sur le principe de la fusion d'états. Cependant, dans le cas des DFA, l'automate A n'est pas initié avec le MCA mais avec le PTA (Prefix Tree Acceptor) qui est en réalité le MCA ayant subit des fusions pour déterminisation. De même la fonction de fusion d'états est suivie par une fonction de fusion pour déterminisation de l'automate, afin qu'il reste un DFA.
La différence entre RPNI et EDSM se situe au niveau de la fonction *choix_de_deux_etats* (A, q_1, q_2). Cette fonction choisit deux états, q_1 et q_2, de A qui seront fusionnés par l'algorithme. Si aucune fusion n'est plus possible, la fonction retourne faux.
La fonction *choix_de_deux_etats*() dans l'algorithme RPNI choisit les couples d'états à fusionner selon un ordre prédéterminé standard, ce qui lui donne ses propriétés d'identification (Higuera et coll., 1996). EDSM, par contre, choisit à chaque étape le couple d'états qui, par fusion déterministe, engendre la fusion d'un maximum d'états finals. Il s'agit intuitivement de fusionner les plus grands suffixes commun. Cette heuristique a permis à l'algorithme EDSM de gagner la compétition d'inférence grammaticale Abbadingo (Lang et coll., 1998a).
Dans le même article de P. Dupont, L. Miclet et E. Vidal (Dupont et coll., 1994), l'espace de recherche pour automates non déterministes par fusion d'états est également décrit. Bien qu'aucun algorithme pour l'inférence de NFA n'avait été proposé dans ce contexte, nous citons ici quelques références. Il s'agit en premier d'un algorithme d'inférence de NFA dans MAT (Angluin, 1987), il est dû à Yokomori (Yokomori, 1994). Le deuxième infère non pas des NFA, mais des fonctions rationnelles pouvant être représentées dans certains cas par des NFA (Bergadano et Varricchio, 1996). Le troisième est apparu en 2000 afin d'inférer une nouvelle sous classe des NFA, la classe des RFSA (Denis et coll., 2001).
Enfin, nous allons présenter en (2.4), comment les travaux de D. Fredouille (Fredouille, 2003) ont fait émerger de nouveaux algorithmes d'inférence de NFA fonctionnant sur le principe de la fusion d'états.

2.4 Premiers algorithmes d'inférence de NFA appliqués à la biologie

Cette thèse s'est bâtie à partir de différents travaux menés précédemment dans l'équipe Symbiose de l'Irisa. Les premiers essais d'apprentissage par heuristique de fusion d'états ayant donné des résultats intéressants sur ce type d'application ont été menés lors du stage de B. Idmont dans l'équipe Symbiose (Idmont, 2002) et ont permis de montrer que les résultats obtenus étaient meilleurs que les mêmes expériences menées en utilisant les algorithmes classiques d'apprentissage de DFA comme RPNI et EDSM. L'approche a introduit l'utilisation et la fusion de fragments générés par le programme Dialign (Morgenstern, 1999).

D'autres travaux ont été orientés par D. Fredouille (Fredouille, 2003) vers un apprentissage de NFA typés (avec apport de connaissances sémantiques). Mais les applications sur des NFA non-typés ont été assez difficiles.
En effet, le constat établi par D. Fredouille au sujet des NFA non-typés lors d'une expérience d'apprentissage sur des protéines alkaline phosphatase (Fredouille, 2003) était le suivant : "Ces automates obtenus sans typage sont trivialement trop généraux. Ils n'intègrent aucunement ne fusse que la notion de longueur des exemples [...] Le problème posé ne semble pourtant pas difficile : les séquences fournies sont en effet très similaires, courtes et peu nombreuses".
De même il conclut au sujet d'une expérience d'apprentissage d'automate sur des protéines MIP : "L'heuristique a montré ses limites sur des véritables données biologiques".
En conclusion, les premiers essais nous ont donc permis de constater que ces approches étaient encourageantes mais non-utilisables à ce stade.
De son côté, A. Leroux (Le Roux, 2005) a également développé une méthode d'inférence dite SDTM calculant des alignements locaux entre paires de protéines et produisant des machines séquentielles proche de celle de Mealy. La base théorique reposant cette fois-ci sur des automates à transition (FTA définis en 2.13) Là aussi plusieurs problèmes ont été soulevés au moment de la confrontation à des données réelles.
Voici les difficultés auquelles les différents auteurs précédents ont tous été confrontés et dont la résolution semble nécessaire avant de pouvoir disposer d'algorithmes d'apprentissage de NFA sur des jeux de protéines :

- surgénéralisation
- perte des domaines identifiables trivialement
- l'approche par préfixe ou suffixe ne semble pas adaptée à l'application
- perte de la structure par fusions engendrant de grandes boucles

2.4.1 Conclusion

Ce chapitre nous a permis de poser les bases théoriques de l'apprentissage d'automates sur des séquences biologiques. Il a permis également de situer ce type de modèles dans la hiérarchie de Chomsky. Dans le chapitre 4 nous allons voir, au travers de notre nouvelle approche, différents algorithmes d'apprentissage de NFA inspirés des précédents travaux connus dans le domaine. Nous préciserons les différentes adaptations qui nous ont été nécessaires avant de pouvoir obtenir des résultats intéressants comme ceux présentés dans la partie III.

Deuxième partie

Apprentissage d'automates non-déterministes sur des familles de protéines

Nous exposons dans cette partie une approche heuristique permettant d'apprendre des NFA à partir de jeux de séquences de protéines. Cette approche est innovante par le niveau d'expressivité recherché pour le modèle. En effet, les NFA peuvent présenter différents chemins d'acceptations et être le support d'une certaine sémantique. L'approche sera ici déclinée en différents algorithmes.
L'idée principale a été de transposer les procédés d'apprentissage par fusion d'états du chapitre 2. Motivés par les premières tentatives effectuées dans l'équipe et se basant sur les perspectives évoquées alors, nous avons abouti à une séparation du problème en deux étapes. La première est une phase d'alignement, dont le but est d'identifier les états candidats à la fusion. La seconde phase est une phase de généralisation proprement dite, où l'on applique les fusions choisies et des post-traitements imaginés grâce à la nature des données.
L'idée de notre nouvelle approche est de produire des modèles capables de représenter toutes les zones nécessaires à la caractérisation d'une famille de protéines, en s'affranchissant des alignements globaux, en apportant de la sémantique, tout en conservant de bonnes capacités de reconnaissance. Les modèles que nous souhaitons obtenir ont un rôle se situant dans la continuité des approches exposées au chapitre 1. Il s'agit donc de prendre en entrée des séquences biologiques membres d'une famille et de construire automatiquement un modèle capable de représenter la famille. L'apport des NFA appris sur ces séquences doit provenir du passage à un niveau plus élevé d'expressivité par rapport aux modèles de types alignements, motifs, etc.
Voyons à présent au travers des deux phases distinctes (alignement et généralisation) les réponses apportées dans la construction d'un modèle de type automate.

Chapitre 3

Alignement Multiple Partiel et Local

Ce chapitre aborde la première phase de l'approche, à savoir la production d'un nouveau type d'alignement de positions par analyse d'un jeu de séquences échantillon d'une famille de protéines. Ce nouveau type d'alignement correspond tout à fait au type de modèle que nous souhaitons obtenir, à savoir des automates qui permettent des compositions de zones similaires.

3.1 *Partial local multiple alignement* (PLMA)

A partir de séquences de protéines, nous cherchons à mettre en valeur les caractéristiques communes qui en font une famille. Certaines familles de forte similarité entre les séquences permettent une caractérisation par un alignement global des positions. Cependant d'autres familles présentant des domaines espacés témoignent d'une caractérisation faite de plusieurs alignements locaux. C'est une séparation que l'on qualifie d'horizontale, qui met en valeur différents domaines de quelques positions. Enfin, nous avons pu remarquer qu'il existe aussi des similarités partagées par quelques groupes de séquences qui forment alors des sous-familles. Nous qualifions ces séparations de verticales, car créant des sous-ensembles des séquences habituellement superposées par les alignements globaux
Afin de mettre en valeur les différentes séparations verticales et horizontales dans un jeu de séquences, nous cherchons donc à produire un alignement multiple qui soit à la fois local et partiel. En pratique ce type d'alignement revient à mettre en relation des positions appartenant aux séquences du jeu représentant la famille d'intérêt. Nous utilisons le terme d'alignment (définie en 3.1) un ensemble de positions reliées au sein d'un jeu de séquences. L'ensemble des associations relevées forme alors une partition (définie en 3.2). Notons qu'il est possible que certaines positions ne participent à aucune relation de similarité. Classiquement, un alignement multiple est une partition du jeu de séquences respectant une contrainte d'ordre entre les positions de chaque séquence. Notre méthode consiste à rechercher des blocs dits blocs de PLMA (définis en

3.8), puis à les assembler selon différentes contraintes. Le résultat obtenu est un PLMA (défini en 3.9) pour Partial Local Multiple Alignment. Un PLMA étant un ensemble de blocs de PLMA eux-mêmes constitués d'ensemble d'associations de positions, alors nous pouvons représenter un PLMA sous la forme d'une partition. Il est important ici de bien différencier la notion de POA (Alignement Partiel Ordonné) qui ne représente que des associations ordonnées de positions, et la notion de PLMA. L'association de positions dans un PLMA est basée sur un contexte et a une valeur sémantique importante. Par sémantique nous entendons le fait de pouvoir trouver des zones similaires qui peuvent correspondre à des domaines, des structures, des enchaînements de propriétés physico-chimiques, ou bien des sites actifs particuliers.

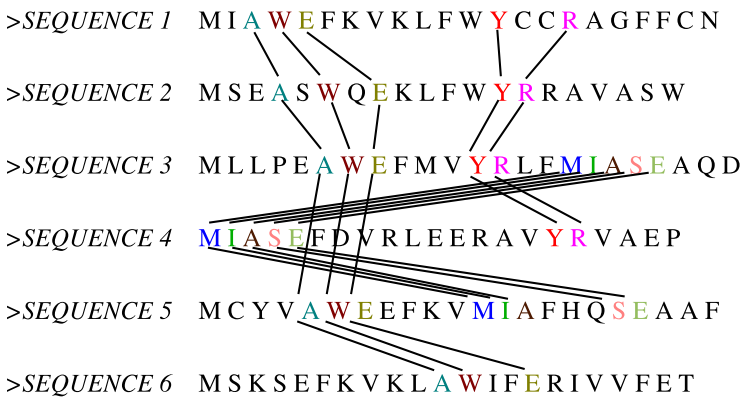


FIG. 3.1 – Partitionnement des positions d'un ensemble de séquences.

Définition 3.1 *Soit S un ensemble de séquences et notons $S_{i,j}$ la position j de la séquence i. Un alignement (ou partie) A_x est un ensemble de position $A_x(S) = \{S_{i_1,j_1}, S_{i_2,j_2}, ..., S_{i_n,j_n}\}$ tel qu'il existe un lien direct, ou par transitivité, de similarité entre toutes les positions de l'alignement.*

Définition 3.2 *Soit S un ensemble de séquences et $S_{i,j}$ représente la position j de la séquence i. Une partition P_x est un ensemble d'alignements. Formellement, un ensemble E d'alignements S est une partition de S si :*

- *Aucun élément de E n'est vide*
- *L'union des associations est égal à l'ensemble des positions de S*
- *Les associations sont deux à deux disjointes*

3.2 Similarité

Nous avons donc proposé l'obtention d'un PLMA basé sur des relations de similarités comme objectif de la caractérisation. Revenons alors sur ce que l'on entend par similarité et quelles sont les entités comparées dans ces relations de similarités.

En bioinformatique, la notion de similarité entre deux objets de même nature est utilisée à tous les niveaux d'observation du vivant :

- Les espèces sont étudiées par les phylogénistes selon des mesures de similarités (comparaison de caractéristiques physiques par exemple).
- Les cellules peuvent également faire l'objet d'un rapprochement selon des critères de rôle fonctionnel (cellules caractéristiques du fonctionnement d'un organe par exemple).
- Dans le cas de séquences de protéines ou d'ADN, les alignements globaux obtenus permettent de choisir une mesure de similarité obtenue par alignement et utilisation de matrices de substitution.
- Pour la structure 3D des protéines le RMSD est la mesure classique d'évaluation d'éloignement entre deux structures.
- Il existe, à plus bas niveau, de nombreuses mesures de distances entre acides aminés.

Dans notre problématique de caractérisation de familles de protéines, les zones fonctionnelles et similaires sont mises en valeur par un processus d'évolution et de pression sélective. En effet les zones fonctionnelles ou domaines, montrent une faible divergence par rapport aux zones non fonctionnelles.
Si nous nous plaçons sous l'angle de la recherche d'une partition, nous allons aligner des positions. Nous pouvons déjà, pour ce niveau de précision, obtenir des informations sur l'acide aminé correspondant. Rappelons alors qu'entre un acide aminé a_1 à une position p_1 et un acide aminé a_2 à une position p_2 nous pouvons affecter un poids ou une distance ou une similarité $S(a_1, a_2)$ en fonction de matrices existantes. Ces matrices ont pu être construites en fonction de la fréquence de substitution, de propriétés physico-chimiques ou encore de structures tridimensionnelles.
Cependant, nous pouvons constater qu'une position dans une zone correspondant à un domaine n'est pas uniquement caractérisée par l'acide aminé correspondant à cette position, mais aussi au contexte de cette position. Par contexte on entend principalement les positions voisines de la position étudiée.

3.3 Paires de fragments significativement similaires (SFP)

Afin de mettre en valeur des sites actifs ou des structures particulières, nous allons observer les fragments (définition 3.3) de différentes séquences. Nous allons alors introduire des mesures de similarité correspondant à des associations de fragments. Ces associations se feront par paires dans un premier temps, puis par ensembles plus importants dans un second temps. La finalité étant la construction de la partition représentant le PLMA final.

Définition 3.3 *Soit une séquence $S = p_1...p_n$. Un fragment ou sous-séquence F de S est un sous-ensemble de positions contiguës $p_i...p_j$ de S.*

B. Idmont (Idmont, 2002) a proposé l'utilisation des paires de fragments d'un programme initialement utilisé pour obtenir des alignements multiples globaux. Ce pro-

gramme se nomme Dialign2 (Morgenstern, 1999) et dispose d'une option permettant d'obtenir un ensemble de paires de fragments. Ce programme permet de prendre en considération des paires $p = (F_1, F_2)$ de fragments tel que $|F_1| = |F_2| = |p|$ et tel que $|p|$ peut prendre des valeurs différentes. L'avantage étant que la mesure de similarité, par estimation du poids des diagonales, est plus homogène et permet de comparer des diagonales de tailles différentes.
Nous avons introduit le terme de SFP(définition 3.4) pour désigner des paires de fragments significativement similaires. Cette notion repose sur la notion de seuil. Une SFP est donc l'association entre une diagonale de type Dialign, son score de similarité, et un seuil de similarité choisi par l'utilisateur de l'approche.

Définition 3.4 *[SFP (Significantly Similar Fragment Pair)] Une paire de fragments* (F_1, F_2) *est significativement similaire si, pour une fonction de similarité ou un score* $w()$ *et un seuil* t, $w((F_1, F_2)) > t$.

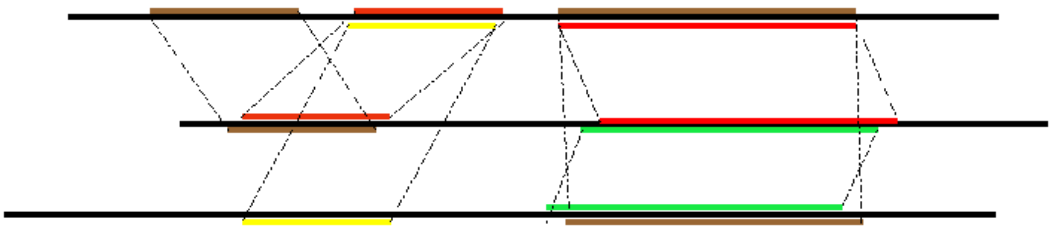


FIG. 3.2 – Représentation de plusieurs SFP dans un jeu de séquences.

De manière générale, nous cherchons à obtenir un score de similarité correspondant dans un premier temps à une comparaison entre deux fragments. Si l'on pose un fragment F_1 et un autre fragment F_2, il est possible de comparer et d'estimer une similarité entre F_1 et F_2, même si $|F_1| \neq |F_2|$. En effet, nous avons vu qu'il existait des algorithmes de type Blast qui permettent d'aligner deux séquences ou fragments de séquences et d'y affecter un score relatif à la qualité de l'alignement (voir chapitre 1).
On peut également choisir d'autres méthodes réduisant le nombre d'associations possibles en imposant une longueur fixe $|F_1| = |F_2|$ comme dans Gemoda (Jensen et coll., 2005).
Cependant le choix de Dialign nous a permis de contrôler la taille de notre espace de recherche (ensemble des SFP). De plus, les stratégies développées dans Dialign telles que le choix de ne pas considérer comme alignées certaines positions ne participant pas à des diagonales, nous ont confortés dans l'idée de chercher à se focaliser sur l'alignement de zones caractéristiques précises, localisées et ne recouvrant potentiellement qu'une partie des séquences.

3.4 Evaluation du score d'une SFP

Nous souhaitons un score de similarité de SFP suffisamment fin pour permettre à l'utilisateur de paramétrer ce qu'il considère comme seuil de significativité. De plus, un score homogène nous permettra éventuellement de choisir entre deux SFP en relation d'incompatibilité (nous détaillons les contraintes possibles en 3.9). L'approche de Dialign nous semblant particulièrement appropriée, nous avons choisi le score de Dialign pour évaluer le poids d'une SFP (voir section 3.3). Voici la façon dont ce score est calculé :

- la définition 3.5 de la similarité d'une SFP
- la façon de calculer le score d'une SFP dans la définition 3.6

Définition 3.5 *Etant donnée une SFP P comprenant un fragment F_1 et un fragment F_2, une matrice B de substitution (exemple Blosum), la similarité de $P(F_1, F_2)$ est la somme $\Sigma_x^{|P|} B(F_{1,x}, F_{2,x})$ avec $F_{i,x}$ l'acide aminé à la position x sur le fragment i.*

Définition 3.6 *Etant donné une SFP (F_1, F_2), le poids $w(F_1, F_2)$ est égal à $-logP(s, l)$ tel que :*

- *$P(s, l)$: la probabilité pour une paire de fragments aléatoire de longueur l d'avoir une similarité supérieure à s*
- *s : la similarité de (F_1, F_2)*
- *l : la longueur de F_1 et F_2*

3.5 Représentation des SFP sous forme d'un graphe de fragments

De façon pratique, l'ensemble des SFP peut être représenté par un graphe. Chaque sommet est un fragment et les arêtes correspondent aux liens introduits par la mise en relation de deux fragments au sein d'une SFP. Le graphe est alors valué en pondérant chaque arête par l'évaluation du score de la SFP correspondante.
Soit S le jeu de protéines à caractériser (figure 3.3). Sachant que l'on peut décomposer chaque séquence Q de S en $\frac{|Q|(|Q|+1)}{2}$ fragments de positions et de tailles différentes (figure 3.4 et figure 3.5), notre approche produit alors un graphe réduit à l'ensemble des fragments participant à des SFP. On construit le graphe $G = (V, E)$ dont l'ensemble des sommets V représentent les fragments F_x de S (figure 3.6), dont les arêtes E sont valuées par la similarité entre les paires.

Définition 3.7 *Une SFP (Significantly Similar Fragment Pair) est équivalent représentée par une arête E dans le graphe des fragments G.*

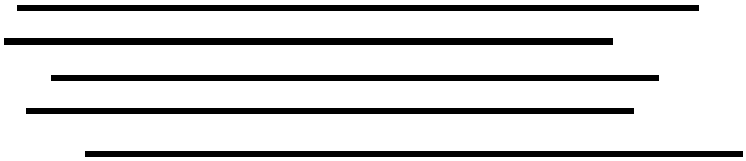


FIG. 3.3 – Le jeu de séquences S

FIG. 3.4 – Décomposition d'une séquence en fragments

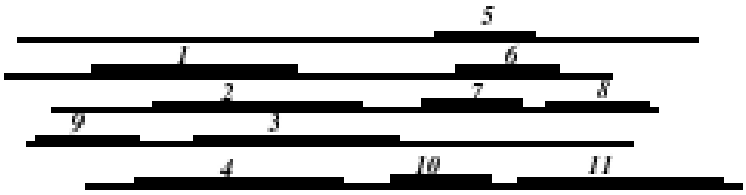


FIG. 3.5 – Représentation de plusieurs fragments dans le jeu S

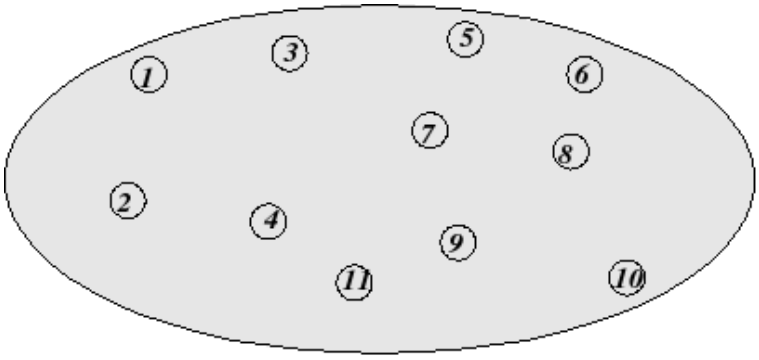


FIG. 3.6 – Sommets du Graphe G des fragments

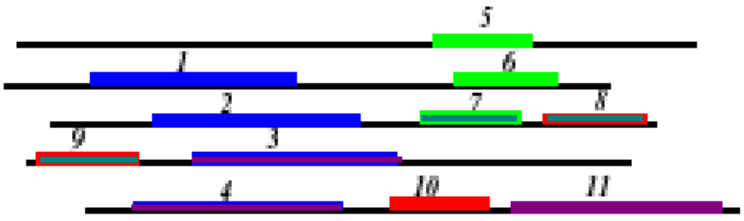


FIG. 3.7 – Identification de fragments significativement similaires

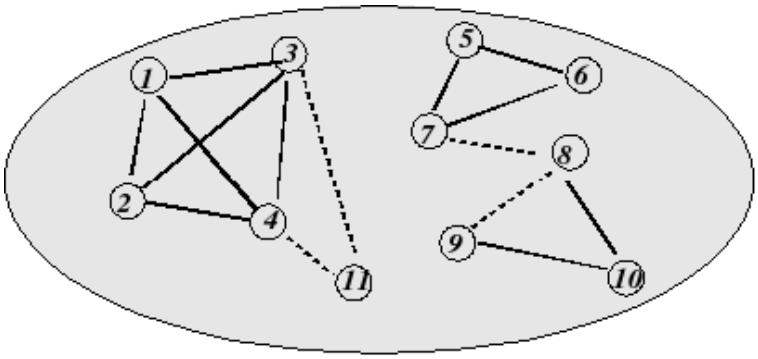


FIG. 3.8 – Graphe G des fragments impliqués dans des SFP représentés par les arêtes

3.6 Bloc de PLMA

Nous posons donc que les SFP ont été produites à partir des paires de fragments de Dialign. Nous cherchons à mettre en évidence des zones s'étendant sur plusieurs séquences. Nous avons donc cherché à ce niveau à étendre la notion de SFP à des ensembles comportant plusieurs fragments liés entre eux par une similarité significative que nous nommons Blocs de PLMA (définition 3.8).
Un bloc de PLMA est un ensemble de fragments et donc de positions alignables contiguës et sur plusieurs séquences. Le Bloc de PLMA est un ensemble qui représente ce que nous souhaitions identifier, à savoir des zones similaires localisées et pouvant témoigner de domaines biologiques, de sites actifs, ou tout simplement d'un ancêtre commun proche sur le plan de l'évolution. Intuitivement les Gaps (voir chapitre 1 ne font pas partie des Blocs de PLMA, mais sont plutôt des zones neutres, non informatives, permettant de produire dans une structure fonctionnelle de la protéine, les enchaînements de ces Blocs. Un Bloc de PLMA (défini en 3.8) est donc un ensemble de fragments significativement similaires qui attestent d'une zone caractéristique dans un jeu de séquences biologiques.

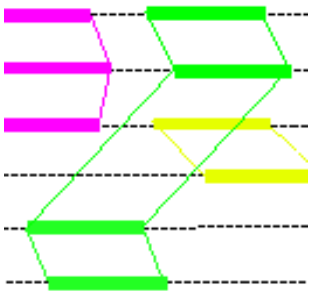


FIG. 3.9 – Représentation de plusieurs Blocs de PLMA. En pratique, notre approche utilise des fragments contigus et de même taille.

Définition 3.8 *Soit un ensemble de fragments* $E = \{F_1, ..., F_n\}$*, E est un Bloc de PLMA si* $\forall (F_x, F_y) \in E$*,* $\exists SFP(F_x, F_1)...SFP(F_j, F_{j+1})...SFP(F_n, F_y)$*.*

Définition 3.9 *Un PLMA (Partial Local Multiple Alignment) est un ensemble de Blocs de PLMA lié à un jeu de séquence. Il peut être représenté par une partition du jeu de séquence.*

3.7 Evaluation du score d'un bloc de PLMA

A la différence des paires de fragments, nous ne disposons pas de mesure homogène capable d'évaluer un score de similarité pour un Bloc de PLMA donné. C'est pourquoi, par la suite nous utilisons des méthodes classiques telles que le score SOP (Sum of Pairs) : $w_{bloc}(b) = \sum_{(F_1,F_2)\in b} w_{sfp}(F_1, F_2)$.
En se basant sur un fragment F appartenant à un bloc b, on peut également choisir un score en étoile : $w_{bloc}(b) = \sum_{F'} w_{sfp}(F, F')$ tel que $(F, F') \in b$.

3.8 Construction de blocs de PLMA à partir du graphe des fragments

Nous disposons d'une représentation de l'ensemble des SFP sous la forme d'un graphe de fragments (voir 3.5).
Puisque les Blocs de PLMA sont des ensembles de fragments liés par une relation de similarité (définition 3.8), alors nous pouvons établir que la construction d'un bloc de PLMA à partir du graphe des fragments revient à choisir une composante connexe (définition 3.10) dans le graphe des fragments.

Définition 3.10 *Une composante connexe CC dans un graphe G est un ensemble de sommets tel que pour tous sommets* S_1, S_2 *de CC, il existe un chemin C ayant comme extrémité* S_1 *et* S_2 *et ne passant que par des sommets de CC.*

La recherche de blocs de PLMA à partir du graphe des fragments représentant les SFP revient donc à un problème de recherche de composantes connexes dans le graphe des fragments.
Nous faisons la remarque suivante : il suffit d'une seule SFP (F_1, F_2) pour lier un fragment à un bloc de PLMA contenant F_2 et obtenir un nouveau bloc de PLMA de taille plus importante. Cette méthode permet d'introduire de la transitivité dans les relations entre les fragments. Elle produit des blocs de PLMA que nous qualifions alors de consensus faible.

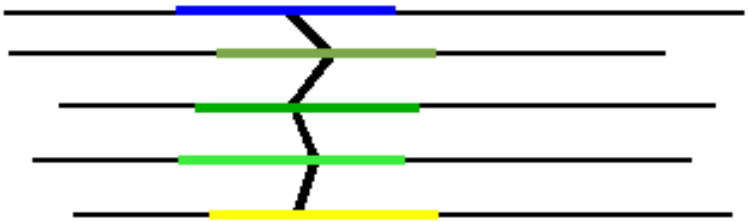


FIG. 3.10 – Le consensus faible représenté par la transitivité dans un Bloc de PLMA classique (composante connexe)

Cependant, compte tenu du contexte biologique de certaines familles, un PLMA se doit d'exhiber un consensus fort. En effet là où la transitivité d'une composante connexe risquerait d'associer des fragments n'ayant rien en commun, nous choisissons alors d'utiliser la notion de clique (définition 3.11) plutôt que de composante connexe et ainsi forcer les Blocs de PLMA à représenter des consensus forts (figure 3.11).

Définition 3.11 *Une clique CL dans un graphe G est un ensemble de sommets tel que pour tous sommets S_1, S_2 de CL, il existe une arête reliant S_1 à S_2.*

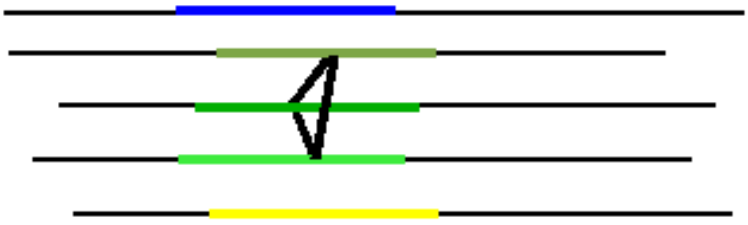


FIG. 3.11 – Le consensus fort représenté par un Bloc de PLMA restreint à une clique

3.9 Incompatibilités entre SFP

Certes, l'identification des informations est un problème majeur auquel nous avons été confronté. Cependant, les premières expérimentations (Idmont, 2002) nous ont montré que les zones similaires étaient parfois superposées. Ce type de situation apparaît

notamment lorsque l'on cherche à traiter de nombreux SFP de faible similarité, mais très présentes dans la famille. Malheureusement, il peut donc arriver que ces informations considérées comme faiblement importantes viennent perturber les alignements et diminuer la qualité des résultats en parasitant les SFP de forte similarité trouvée précédemment.
Selon les types de PLMA que l'on veut autoriser, nous avons alors choisi d'identifier différentes contraintes entraînant des incompatibilités entre SFP.
Dans un premier temps nous avons introduit la notion de Préservation de fragments (définition 3.12). C'est une contrainte qui permet, une fois un fragment identifié comme informatif, de préserver les positions de ce fragment en empêchant qu'elles puissent être associées les unes aux autres.

Définition 3.12 *Soit un ensemble de séquences S. Soit P la partition représentant $PLMA(S)$. Soit B l'ensemble des blocs de PLMA de $PLMA(S)$. Alors $\forall F \in B$, $\nexists p_i, p_j$ associés dans P avec $p_i \in F$ et $p_j \in F$.*

Dans un second temps, nous avons remarqué que ce qui semblait trivial pour un alignement multiple a dû être repensé dans le cadre de notre modélisation.
En effet sans l'introduction de contraintes, notre démarche permet par exemple les associations entre positions d'une même séquence. Cependant, il faut savoir que ceci introduirait des répétitions dans les modèles de type automate que nous produirions par la suite.
De même, à l'image du principe des diagonales de Dialign nous avons souhaité disposer une contrainte permettant de ne pas introduire de croisements entre les positions ordonnées d'une même séquence par association avec des positions en ordre différent sur une autre séquence ou par le jeu des différentes associations entre différentes séquences. Ces inconsistances sont alors évitées par une contrainte dite de Consistance 3.13. L'étude de Dialign nous a permis de connaître et d'utiliser une librairie nommée Gabios (Abdeddaïm et Morgenstern, 2000) qui traite rapidement ce type de contraintes.

La contrainte d'inconsistance est violée, lorsque deux positions d'une même séquence se trouvent associées à deux positions en ordre inverse sur une autre séquence.

Définition 3.13 *Soit un ensemble de séquences S. Soit P la partition représentant $PLMA(S)$. Soit $S_{i,s}$ la position i de la séquence s. La contrainte de Consistance pour $PLMA(S)$ est violée si :*
$\exists\ A_1, A_2 \in P$ *tel que* $\exists S_{x,a}, S_{y,b} \in A_1$, *et* $S_{k,c}, S_{l,d} \in A_2$, *avec* $a = b$, $c = d$,
$x < y$ *et* $k > l$

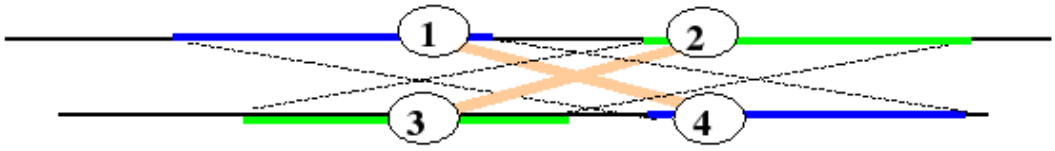


FIG. 3.12 – Violation in de la contrainte de Consistance entre les positions 1 et 2 associées à la position 3 et 4..

3.10 Incompatibilités entre blocs de PLMA

Au delà des incompatibilités entre SFP, nous avons pu montrer qu'il existait des situations mettant en interférence des Blocs de PLMA (Coste et Kerbellec, 2006). En effet, la découverte d'un Bloc de PLMA correspond à l'identification, la caractérisation et l'association de plusieurs positions. Puisque nous caractérisons ces positions par rapport à un contexte particulier, nous choisissons alors d'éviter que l'information identifiée à une position ou dans un contexte particulier soit altérée. Le fait qu'il y ait une intersection non vide entre deux Blocs de PLMA peut participer à cette altération par l'unification de groupes de positions liés par une petite similarité entre deux positions extrêmes.
Nous avons donc cherché à conserver l'invariant suivant : on ne peut aligner des positions qui ne participent pas, dans le cadre des fragments auxquels elle appartiennent, à une même composante connexe ou à une même clique dans le graphe des fragments. Une situation d'interférence (définition 3.14) de Blocs de PLMA est apparente lorsque deux positions se trouvent liées par la superposition de plusieurs Blocs de PLMA, et qu'il n'existe pourtant aucun SFP ou Bloc de PLMA qui confirme cette association.

Définition 3.14 *Soit un ensemble de séquences S. Soit P la partition représentant $PLMA(S)$. Soit p_1, p_2 des positions dans le jeu S. Soit EB l'ensemble des Blocs de PLMA. La contrainte de Non-Interférence pour $PLMA(S)$ est violée si :*
$\exists A_1 \in P$, *tel que,* $\exists p_1, p_2 \in A_1$ *et* $\nexists p_1, p_2 \in EB$

FIG. 3.13 – En cas d'interférence il faut choisir entre les deux Blocs de PLMA

3.11 Problème : trouver le meilleur PLMA compatible

Nous avons donc établi que la phase d'alignement avait pour objectif l'obtention d'un PLMA. D'autre part, nous avons vu qu'un ensemble de Blocs de PLMA permet d'obtenir un PLMA par l'identification des liens de similarités entre positions. Enfin, le fait d'avoir introduit des scores permettant d'évaluer et de comparer des SFP ou des Blocs de PLMA nous permet de choisir lorsqu'il y a des situations d'incompatibilité entre différentes entités.

Etant donné un ensemble de différentes contraintes de compatibilité, nous pouvons alors définir un PLMA compatible (Coste et Kerbellec, 2007) :

Définition 3.15 *Un PLMA compatible est un PLMA produit par un ensemble de blocs de PLMA compatibles et par un ensemble de SFP respectant également des contraintes de compatibilité.*

Nous pouvons alors résumer notre problématique concernant la phase d'alignement de la manière suivante, afin de montrer que l'on peut tenter d'y répondre par des outils classiques de la théorie des graphes :

- Le but est de former le meilleur PLMA compatible à partir d'un jeu de séquences
- Ce PLMA est obtenu par la production d'un ensemble de Blocs de PLMA
- Ces Blocs de PLMA doivent être compatibles
- Chaque Bloc de PLMA est composé d'un ensemble de SFP également compatibles
- Lorsqu'il y a incompatibilités, on choisit l'ensemble des Blocs de PLMA qui maximise un score total de similarités

Le problème de notre alignement d'un jeu de séquences S peut donc être formalisé dans la définition 3.16.

Définition 3.16 *Posons G le graphe des fragments de S,*
$w_{bloc}()$ une mesure de similarité de Bloc de PLMA,
un Bloc de PLMA b qui est représenté par CC_b (composante connexe de G),
Posons la fonction $EstCompatibleE_b$ comme vérifiant la contrainte de compatibilité d'un ensemble de blocs.
Alors $PLMA(S) = \{CC\_i, ..., CC\_j\}$ vérifiant $EstCompatible()$.
Nous recherchons donc la fonction C d'alignement d'un jeu de séquences S tel que $C(S) = Argmax(\sum_{x=i}^{x=j} w_{bloc}(CC_x))$.

3.12 Algorithmes

Après cette étape de formalisation, nous allons à présent étudier différents algorithmes que nous avons choisis pour résoudre ce problème. Nous commencerons en posant le cadre général, puis les 2 premières tentatives et enfin l'approche finale véritablement retenue.
Le schéma général apparaît dans ce manuscrit afin de montrer la trame empruntée par les différentes solutions. La première solution, appelée *méthode strictement SFP* a été

implémentée et testée sous le nom de Protomata-L (Coste et coll., 2004). Puis c'est la méthode dite *clique semi-exhaustive* qui a fait l'objet d'une publication sous le nom de Protomata-CL (Coste et Kerbellec, 2005). La version courante, par graines de bloc de PLMA, est depuis évoquée sous le nom de Protomata-Learner (Coste et Kerbellec, 2006) et fait l'objet de constantes évolutions comme les différents types de consensus évoqués ici.

3.13 Schéma général de l'approche

La taille du graphe des fragments dépend du nombre de SFP trouvés dans le jeu de séquences et donc du seuil de similarité choisi par l'utilisateur. Cependant le nombre de Blocs de PLMA et les combinaisons d'ensembles de Blocs de PLMA compatibles pour former un PLMA final deviennent rapidement complexes à évaluer de façon exhaustive. C'est pourquoi nous avons choisi de nous baser sur un algorithme glouton classique. Cette première approche de type *best first* nous permet d'effectuer des premiers tests dans des temps raisonnables afin de stabiliser l'approche et de transférer par la suite notre formalisation vers des domaines plus efficaces telle que la recherche opérationnelle. Pour notre approche de type *best first*, il s'agit de générer l'ensemble des Blocs de PLMA puis de les ordonner selon le score $w_{bloc()}$. Puis chaque Bloc de PLMA est dépilé par ordre décroissant et ajouté à l'ensemble résultat (le PLMA) si et seulement s'il satisfait aux fonctions de compatibilités.

Algorithme 2 Schéma général de la phase d'alignement

Entrées jeu de séquences de protéines S
Sorties ensemble E_B des Blocs de PLMA formant le PLMA choisi pour la caractériser de S

```
E_B ← ∅
G ← CreerGraphe(S)
P_B ← ComposantesConnexes(G)
P_B ← PF.Trier(w_bloc())
Pour chaque B ∈ P_B faire
  Si EstCompatible(B, E_B) alors
    E_B ← B
    PurgeAretesIncompatibles(B, G)
  Fin si
Fin pour
Retourner(E_B).
```

3.14 Résolution par la méthode Strictement SFP

Cette méthode est historiquement la première que nous avons tentée. La particularité de cette méthode est de considérer des Blocs de PLMA à 2 fragments. La pile P_B des blocs de PLMA possibles et ordonnés est alors simplement la pile des SFP ordonnés

selon $w_{sfp}()$. Nous verrons au chapitre 4 que le score $w_{sfp}()$ a fait l'objet de plusieurs variantes lors de l'implémentation de cette approche Strictement SFP.
La fonction *comp*() permettant de vérifier les contraintes de compatibilité est donc également réduite aux contraintes de préservation, d'unicité et d'inconsistance des SFP.
Cette version assez intuitive, permet de construire progressivement les blocs par agrégation de SFP. Les contraintes de blocs n'étaient alors pas implémentées. Nous verrons par la suite que les expériences ont montré de bons résultats. Cependant, le temps d'exécution n'étant pas particulièrement intéressant, il nous a semblé plus rapide d'aller chercher directement les blocs par d'autres méthodes. Cela nous a été en partie confirmé sur cette approche lors de l'introduction de score permettant de pondérer les SFP en fonction de leur support (définition 4.2) ou de leur implication (définition 3.18). Le support permettant de rendre compte de la présence d'une SFP sur l'ensemble de la famille. Et l'implication est une formule normalisée du support d'une SFP dans une famille en prenant en compte son implication dans d'autres jeux de séquences (contre-exemples).

Algorithme 3 Algorithme de caractérisation par méthode Strictement SFP

Entrées jeu de séquences de protéines S
Sorties ensemble E_{SFP} des Blocs de PLMA choisis pour la caractérisation de S

```
E_SFP ← ∅
G ← CreerGraphe(S)
E_SFPtemp ← G.BlocsDePlmaDeTaille(2)
P_SFP ← E_SFPtemp.EvaluerTrier(w_bloc())
Pour chaque CC ∈ P_CC faire
  Si EstCompatible(CC, E_CC) alors
    E_CC ← CC
    PurgeAretesIncompatibles(CC, G)
  Fin si
Fin pour
Retourner(E_CC).
```

Définition 3.17 *Une SFP* (F_1, F_2) *est dite* supportée *par un fragment* F *si :* $w(F, F_1) + w(F, F_2) \geq w(F_1, F_2)$.

Définition 3.18 *Pour chaque SFP* p, *l'*indice d'implication *est basé sur (Lerman et Azé, 2004) et traduit par* $\iota(p)$ *:* $\iota(p) = \frac{-P(Support_N(p)) + P(Support_S(p)) \times P(N)}{\sqrt{P(Support_S(p)) \times |N|)}}$ *où* $|X|$ *renvoie la cardinalité d'un jeu* X, *et* $P(X)$ *est sa proportion par rapport à* S *et* N *:* $P(X) = \frac{|X|}{|S|+|N|}$.

3.15 Résolution par méthode Clique Semi-Exhaustive

Cette version a été choisie au moment où nous avons choisi pour la première fois d'évaluer le comportement de l'approche lorsque l'on souhaite obtenir des Blocs de

PLMA de consensus fort. La particularité de cette version est de procéder par taille de clique décroissante. En effet, nous n'évaluons pas d'un coup l'ensemble des cliques possibles dans le graphe des fragments, ce qui serait trop coûteux. Mais nous choisissons d'énumérer l'ensemble des cliques possibles représentant des Blocs de PLMA couvrant l'ensemble des séquences, puis l'ensemble des séquences moins un, etc.
Chaque niveau est assez complexe à évaluer mais permet un grand élagage pour le niveau suivant grâce aux règles de compatibilités. La priorité a ici été donnée à la taille des blocs.

Algorithme 4 Algorithme de caractérisation par méthode en Clique Semi-Exhaustive

Entrées jeu de séquences de protéines S
Sorties ensemble E_{CL} des Blocs de PLMA choisis pour la caractérisation de P

```
E_CL ← ∅
G ← CreerGraphe(S)
N ← S.NombreDeSequences()
Tant que N > 0 faire
  E_CLtemp ← G.RechercheCliques(N)
  P_CL ← E_CLtemp.EvaluerTrier(w_bloc())
  Pour chaque CL ∈ P_CL faire
    Si EstCompatible(CL, E_CL) alors
      E_CL ← CL
      PurgeAretesIncompatibles(CL, G)
    Fin si
  Fin pour
  N ← N − 1
Fin tant que
Retourner(E_CL).
```

3.16 Résolution par méthode Graines de Blocs de PLMA

Cette méthode est la plus récente et celle que nous utilisons désormais de manière standard. Nous avons vu que l'approche générale était de type gloutonne heuristique. Mais puisque l'énumération des Blocs de PLMA est également complexe, nous ajoutons un autre niveau de simplification. En effet, la méthode graine permet de trier non plus des Composantes Connexes du graphe des fragments mais directement l'ensemble des fragments.
Le fait de ne pas chercher à trier directement des Blocs, mais plutôt des fragments permet une opération linéaire et non plus quadratique ce qui est un gain considérable en temps de calcul. Le principe étant de disposer d'une mesure efficace corrélant le score d'un fragment à celui de la composante connexe dont il fait partie. La pile traitée par l'algorithme glouton n'est donc plus une pile de Blocs mais une pile de fragments. Pour chaque fragment dépilé, il suffit de s'en servir comme *graine* pour l'étendre à la composante connexe ou à la clique maximale dont il fait partie.

Pour l'ordonnancement de la pile P_F des fragments, nous utilisons la fonction de score w_f relative à chaque fragment F dans G. Ce score est choisi comme un *score en étoile*, il est égal à la somme des $w_{sfp}(F_1, F_2)$ dans E_{sfp} tel que $F = F_1$ ou $F = F_2$.
La fonction *comp*() permettant de vérifier les contraintes de compatibilité comprend les contraintes de préservation, d'unicité et d'inconsistance des SFP, ainsi que les contraintes de préservation des PLMA.

Algorithme 5 Algorithme de caractérisation par méthode de Graines de Bloc de PLMA

Entrées jeu de séquences de protéines S
Sorties ensemble E_{CC} des Blocs de PLMA choisis pour la caractérisation de S

```
E_CC ← ∅
G ← CreerGraphe(S)
P_F ← Sommets(G)
P_F ← PF.Trier(w_f())
Pour chaque F ∈ P_F faire
  CC ← ChercheComposanteConnexe(F, G)
  Si EstCompatible(CC, E_CC) alors
    E_CC ← C
    PurgeAretesIncompatibles(CC, G)
  Fin si
Fin pour
Retourner(E_CC).
```

Notons, que nous utilisons également le pendant de cet algorithme dans une version orientée consensus fort. Il suffit de remplacer la fonction $ChercheComposanteConnexe()$ pour une fonction $ChercheClique()$. La démarche est donc identique sauf que lorsque l'on dépile un fragment, c'est la clique maximale à laquelle il appartient que l'on va tenter d'ajouter à l'ensemble solution.

3.17 Conclusion

Nous avons donc posé les bases d'une méthode permettant de produire des alignements nommé PLMA particulièrement adaptés à l'objectif de modélisation des familles de protéines par automates. Cette méthode a vu apparaître les concepts de Blocs de PLMA, ainsi que des contraintes associées à ces blocs ou encore aux différentes SFP qui nous ont permis de trouver ces Blocs sous forme de composante connexe ou de clique dans un graphe de fragments.
La modélisation à partir d'un ensemble de fragments permet de poser le cadre général dans lequel nous évoluons. L'implémentation classique par algorithme glouton et Blocs de PLMA avait pour but l'obtention de solutions performantes pour des temps d'exécution raisonnables. C'est ce que montreront les expériences du chapitre 6.
Les versions orientées *consensus fort* ont également permis d'augmenter les perfor-

mances sur des jeux réputés difficiles.
Cependant, la réalisation d'un algorithme exact pour résoudre ce problème, par programmation linéaire ou autre méthode de recherche opérationnelle, est toujours une question ouverte.

Chapitre 4

Protomates

Ce chapitre présente l'étape de *généralisation* de notre approche. Comme annoncé au chapitre 2, le type de modèle que nous avons choisi pour représenter des familles de protéines est le modèle NFA (automates finis non-déterministes). Toute la difficulté consiste à produire une catégorie de modèles plus expressive que celles utilisées par la plupart des bioanalystes, tout en s'assurant sur le fait que ces modèles présentent des propriétés novatrices et intéressantes, et qu'ils soient à même de scanner des banques de séquences à la recherche de nouveaux membres.
De plus, nous nous sommes basés sur des méthodes d'inférence grammaticale ayant certes montré une certaine efficacité sur des modèles théoriques et des échantillons générés aléatoirement, mais n'ayant que trop rarement eu de succès sur des applications réelles.

4.1 Fusion des positions d'un PLMA

Le chapitre 3 nous a permis d'établir un nouveau type d'alignement nommé PLMA. La production d'un PLMA à partir d'un jeu de séquences est ainsi consacré à fournir une référence (sous forme de partition) à la phase consacrée à l'inférence grammaticale. Cette phase consiste à adapter des outils de l'inférence grammaticale exposés dans le chapitre 2 et plus spécifiquement de l'apprentissage d'automates non déterministes par fusions de fragments. A partir des informations fournies par le PLMA, nous allons en effet pouvoir inférer par application de fusions, un automate modélisant un jeu de séquences.
A partir des concepts d'apprentissage d'automates par fusion d'états introduits en 2.3.3 et des premières expérimentations sur l'apprentissage d'automates non déterministes (Fredouille, 2003; Idmont, 2002), nous allons voir le cadre principal dans lequel va se dérouler la phase de généralisation du langage.
Le principe consiste à créer un MCA (figure 4.1) que nous avons défini en 2.16. Ce MCA représente exactement le jeu de séquences d'apprentissage (figure 4.1).

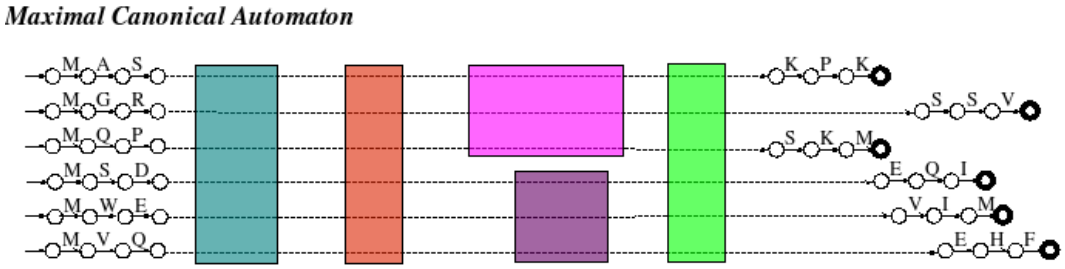


FIG. 4.1 – Identifications des positions associées par le PLMA dans le MCA.

Ensuite, il s'agit d'identifier sur le MCA les positions alignées par la partition correspondant au PLMA obtenu précédemment. Une position correspond à une transition portant un acide aminé, à l'état d'origine et à l'état cible de cette transition. L'association entre deux positions dans le MCA puis dans l'automate courant, va alors se traduire par la fusion d'états (définie en 2.17) prenant en compte la fusion des états d'origines des positions impliquées, ainsi que la fusion des états cibles des positions impliquées.
Remarque : pour des raisons pratiques nous simplifions les notations correspondant à plusieurs transitions entre les deux mêmes états par une seule transition dont la valeur vaut toutes les valeurs existant pour toutes les transitions entre ces 2 états.
Nous pouvons voir un exemple de fusion de positions appartenant à plusieurs fragments significativement similaires sur la figure 4.2. L'opération de fusion entre deux positions est définie en 4.1.

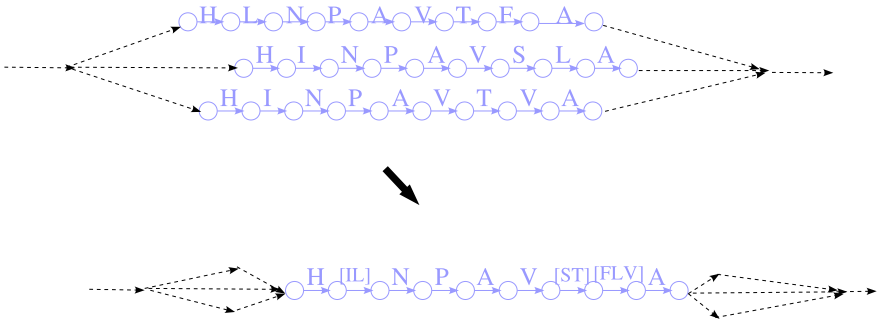


FIG. 4.2 – Fusions des transitions et des états relatifs aux positions similaires de plusieurs fragments.

Définition 4.1 *Soit un automate* $A = \langle \Sigma, Q, I, \delta, F \rangle$ *représentant une famille de séquences.*
On définit une fusion entre deux transitions de A, $t_1 = \langle q_x, a, q_y \rangle$ *et* $t_2 = \langle q_u, b, q_v \rangle$ *comme une nouvelle transition* $t_3 = \langle q_w, C, q_z \rangle$, *tel que* $q_w = fusion(q_x, q_u)$, $q_z = fusion(q_y, q_v)$ *et* $C = \{a, b\}$.

L'opération de fusion est itérée sur toutes les associations fournies par le PLMA original. D'un point de vue apprentissage, le langage représente par le MCA se réduisait au strict jeu de séquences en entrée. Après applications des fusions engendrées par le PLMA, l'automate résultant est capable de reconnaître (d'accepter) de nouvelles séquences. La taille du langage augmente avec les nouveaux chemins créés dans l'automate. On peut observer la fusion des zones significativement similaires à partir de cette première généralisation sur la figure 4.3.

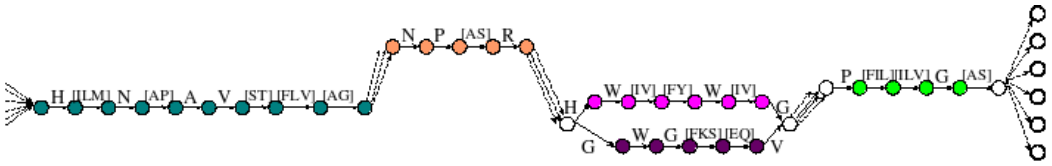


FIG. 4.3 – Fusions des états liés par le PLMA dans l'automate.

4.2 Intérêt des automates à transition

A ce niveau nous faisons la remarque suivante. Bien que les méthodes d'apprentissage par fusion d'état tels que RPNI ou EDSM travaillent sur des automates où les symboles sont situés sur les transitions, nous avons choisi pour des raisons pratiques d'utiliser une représentation par automates à transition (décrits en 2.13).

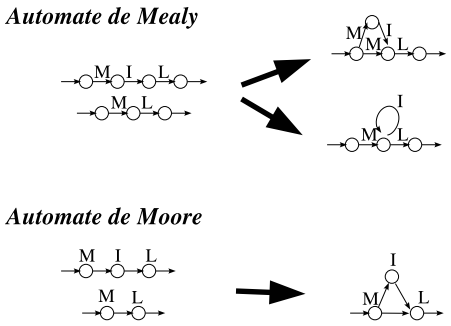


FIG. 4.4 – La fusion d'état sur des automates de Moore se révèle plus pratique que sur des automates de Mealy. On peut ainsi conserver l'information d'une insertion d'un acide aminé, et non créer une boucle ou une répétition de la même position. Ici les position "M" et "L" sont fusionnées.

4.3 Zones caractéristiques

Malgré la généralisation effectuée par les premières fusions, nous pouvons constater que de nombreux états n'ont pas ou très peu participé à des fusions (voir figure 4.1). D'une part, selon le niveau d'observation souhaité par l'utilisateur, ces états peuvent bruiter la mise en valeur des zones constitutives de la fonction. D'autre part la prise en compte des informations relatives à une ou deux séquences uniquement risque de rendre le modèle surspécialisé et de rendre plus difficile la reconnaissance de nouveaux membres de la famille étudiée.
Après avoir défini la notion de support en 4.2, nous introduisons alors un paramètre nommé quorum (défini en 4.3) qui permet de régler le niveau d'observation souhaité.

Définition 4.2 *Soit S un jeu de séquences représentatif d'une famille. Soit un automate $A = \langle \Sigma, Q, I, \delta, F \rangle$ dérivé du MCA comportant $|S|$ automates canoniques représentant les $|S|$ séquences de S. Le support d'un état $q \in Q$ est égal au nombre d'automates canoniques différents dont il dérive.*

Définition 4.3 *Le paramètre* quorum *est un seuil au-dessous duquel le support est considéré comme non caractéristique par l'utilisateur.*

La notion de quorum nous permet alors d'identifier des chemins caractéristiques, constitués uniquement d'états et de transitions marqués comme supportant un nombre de séquences supérieur ou égal au quorum. Nous nommons ces chemins des zones caractéristiques (4.4) et faisons l'hypothèse qu'ils représentent de façon sémantique, une projection de la notion de domaine biologique (ref définition de domaine dans la partie bio) sur les automates.

Définition 4.4 *Soit l'automate $A = \langle \Sigma, Q, I, \delta, F \rangle$ modélisant une famille de séquences. On dit d'un chemin Z dans A qu'il est une zone caractéristique si et seulement si, étant donné un quorum Q : $\forall q_i \in Q$ exercé par Z,* $\text{support}(q_i) >= Q$

4.4 Fusion des états de *gap*

Sur l'automate, obtenu après la fusion du PLMA, nous sommes donc capable d'identifier les zones caractéristiques. Intéressons-nous désormais au traitement des états et des transitions situées entre les zones caractéristiques. Puisqu'il n'y apparaît pas de positions informatives nous avons décidé que ces zones que nous nommons *zones de GAP* et définies en 4.6, pouvaient être fusionnées en un seul état de gap (définition en 4.5). Cet état continue de maintenir les liaisons entre les zones caractéristiques adjacentes, tout en permettant de faire cette liaison par une séquence d'acides aminées de composition non-définie. La figure 4.5 montre la simplification engendrée par les gaps.

Définition 4.5 *Un état de gap e est défini tel qu'il existe une transition $t = \langle e, a, e \rangle$. Si la valeur a de la boucle est égale à l'alphabet Σ.*

Un état de *gap* est l'équivalent du x^* dans les motifs de type Prosite.

Définition 4.6 *Soit l'automate $A = \langle \Sigma, Q, I, \delta, F \rangle$ modélisant une famille de séquences. Soit ZC l'ensemble des zones caractéristiques de A. $\forall E$, un ensemble d'état maximal tel que $\forall e_1, e_2 \in E$, il existe un chemin $C(e_1, e_2)$, tel que $\forall Z \in ZC$, $\forall e_i \in C$, $e_i not \in Z$.*

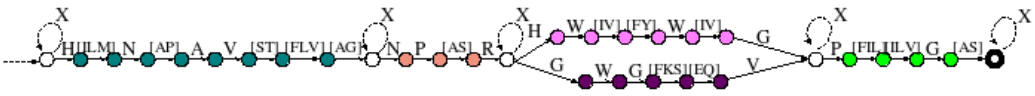


FIG. 4.5 – Automate de protéines avec états de gaps

A cette étape, nous présentons l'algorithme 4.4 permettant d'obtenir un modèle de type automate, appris sur un jeu de séquences biologiques.

Algorithme 6 Approche globale de notre approche d'apprentissage de NFA incluant la fusion du PLMA et les zones de Gap.

Entrées un jeu de séquence S, une fonction de partition $ConstruitPLMA()$, un quorum Q
Sorties un modèle de type automate A
$P \leftarrow ConstruitPLMA(S)$
$A \leftarrow CreerMCA(S)$
$A \leftarrow FusionPositions(A, P)$
$LZ \leftarrow IdentificationZones(A, Q)$
Pour chaque $Z \in LZ$ **faire**
Si $Z \neq Caracteristique$ **alors**
$A \leftarrow FusionGap(A, Z)$
Fin si
Fin pour
$Retourner(A)$.

4.5 Les exceptions

Lorsque l'on travaille sur des familles de séquences réelles, il n'est pas souvent possible de trouver des zones caractérisant d'emblée la famille de protéines dans son ensemble. Dans ce cas, nous avons pu observer que la généralisation des zones de Gap pouvait entraîner des problèmes. En effet, certaines fusions de gaps deviennent des chemins "court-circuits" aux chemins caractéristiques de l'automate. Un exemple, avant généralisation, est donné sur la figure 4.6. Le même exemple, après généralisation, est donné sur la figure 4.7.

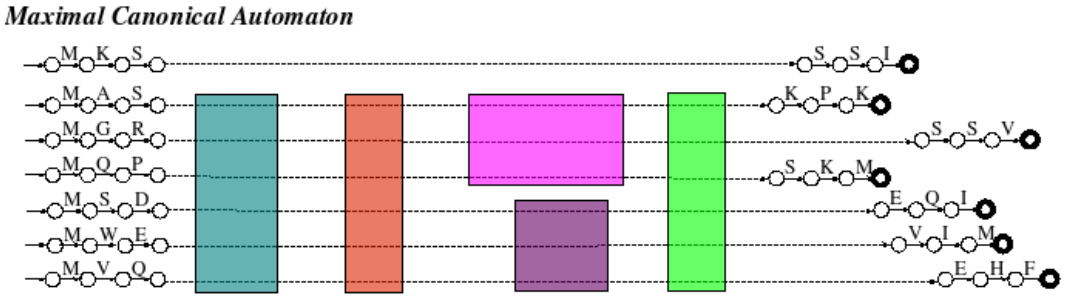


FIG. 4.6 – MCA avec une séquence isolée.

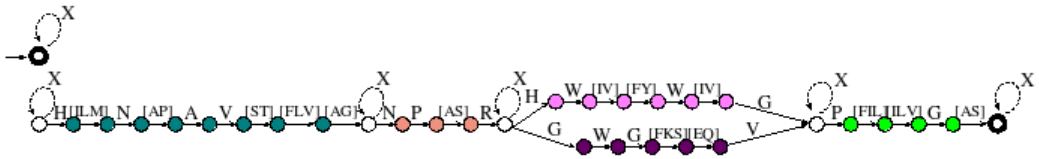


FIG. 4.7 – Automate avec présence d'un automate universel dû au court circuit créé par la fusion en gap d'une séquence exceptionnelle par rapport au reste de la famille.

Nous raisonnons ici dans le but de ne pas perdre des informations précédemment établies par les fusions d'états identifiés par la phase de caractérisation. S'il existe un chemin caractéristique associant toutes les séquences sauf une ou deux exceptions, il n'est pas intéressant de permettre alors la fusion de chemins court-circuits en gap. Cela entraînerait des modèles trop peu spécifiques, puisque permettant un parsing à faible coût par des zones de gaps plutôt que par des zones caractéristiques. C'est pourquoi nous avons introduit la notion de zones Exception (4.7). Les chemins identifiés comme étant exceptions, permettent un marquage d'états ne pouvant participer à la généralisation par fusion en gaps.

Définition 4.7 *Soit l'automate $A = \langle \Sigma, Q, I, \delta, F \rangle$ modélisant une famille de séquences. Soit ZC l'ensemble des zones caractéristiques de A. Soit $Pred()$ la fonction qui pour un chemin dans A renvoie la zone caractéristique précédente, et $Succ()$ la fonction qui pour un chemin dans A renvoie la zone caractéristique suivante. Une zone Exception ZE est un chemin appartenant à une zone de gap ZG et tel que, si l'ensemble des chemins EC dans A, tel que $\exists c, c_1, c_2, c_3 \in EC, c = c_1 \cup c_2 \cup c_3$ et $c_2 \in ZC$, alors $Pred(ZE) = Pred(c_1)$ et $Succ(ZE) = Succ(c_3)$.*

Afin de marquer les états comme faisant partie de chemins exceptions, nous avons eu besoin d'un algorithme suffisamment efficace. Puisqu'il est possible de connaître les zones caractéristiques ainsi que les zones non-caractéristiques situées entre eux,

nous construisons un graphe dont les sommets représentent les zones caractéristiques et les arêtes les connexions établies entre deux zones caractéristiques par une zone non caractéristique. Ce n'est qu'après avoir adapté un algorithme basé sur la fermeture transitive d'un graphe que nous pouvons décider si une zone non-caractéristique est une zone de Gap ou une zone formée de chemins exceptions. En effet, chaque zone non-caractéristique représentée par une arête reliant un sommet S_1 à un sommet S_2 est considérée comme Exception s'il existe un chemin de longueur supérieur à 1 reliant S_1 à S_2. De cette étude nous disposons d'une fonction $f_{trans}()$ permettant de créer le graphe des zones et d'évaluer les longueurs des chemins, et d'une fonction $ExisteCheminTailleSuperieurUn()$ permettant de savoir s'il existe un chemin de taille supérieure à un entre deux sommets.

L'identification des zones Exceptions permet donc à notre approche de produire l'étape de généralisation par gap sans perdre la spécificité recherchée dans les zones caractéristiques. La figure 4.8 montre comment on identifie visuellement une exception. Nous détaillons l'algorithme 4.5 permettant l'identification de tous les chemins exceptions à ne pas fusionner en gap.

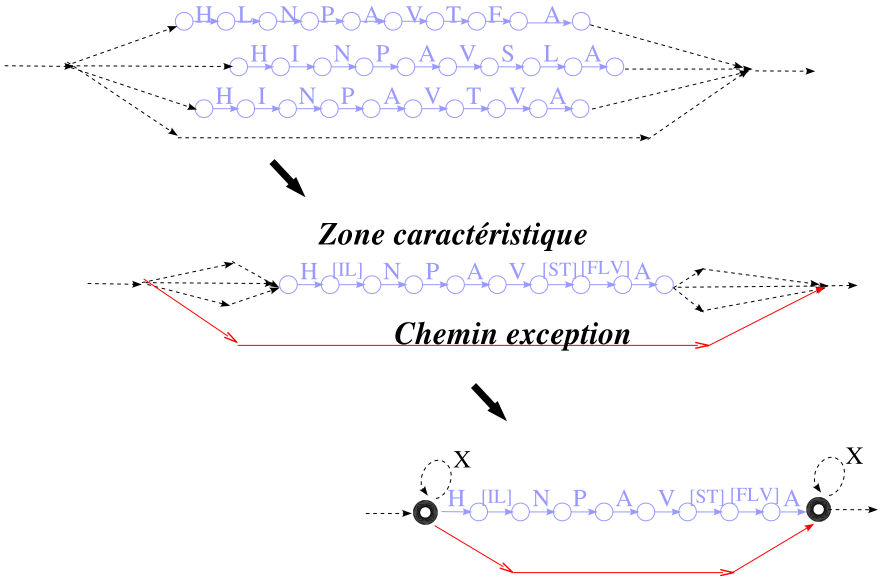


FIG. 4.8 – Identification d'une exception et fusion des zones de gap.

Algorithme 7 Approche par fusion d'états et identification des chemins exceptions

Entrées un graphe vide G, une liste de zones caractéristiques LZ, une liste de zones de Gap LG
Sorties une liste de zones Exception LE et mise à jour de LG

$LE \leftarrow \emptyset$
Pour chaque $Z \in LG$ **faire**
  $G.AjoutSommet(Z)$
Fin pour
Pour chaque $Gap \in LG$ **faire**
  $Z_1 \leftarrow Predecesseur(LG)$
  $Z_2 \leftarrow Successeur(LG)$
  $G.AjoutArete(Z_1, Z_2)$
Fin pour
Pour chaque $Gap \in LG$ **faire**
  $Z_1 \leftarrow Predecesseur(Gap)$
  $Z_2 \leftarrow Successeur(Gap)$
  $Excep \leftarrow G.ExisteCheminTailleSuperieurUn(Z_1, Z_2)$
  Si $Excep$ **alors**
    $LE.Ajout(Gap)$
    $LG.Retire(Gap)$
  Fin si
Fin pour
$Retourner(LE)$

Ainsi, les figures 4.9 et 4.10 montrent l'automate attendu comme résultat de notre approche à partir d'un MCA et d'une phase de caractérisation réussie.

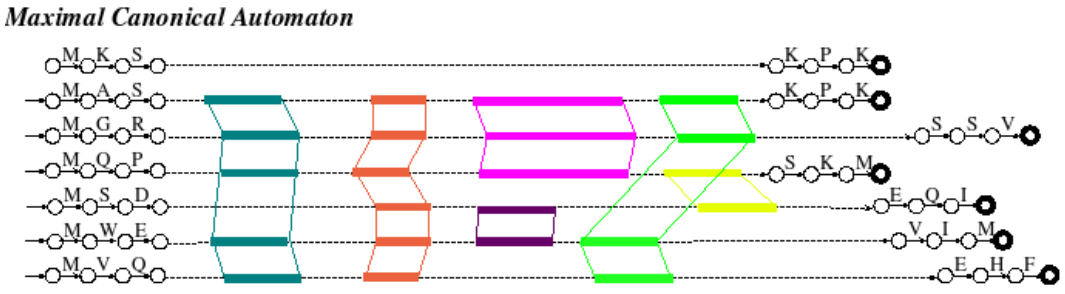


FIG. 4.9 – MCA et mise en valeur des zones similaires.

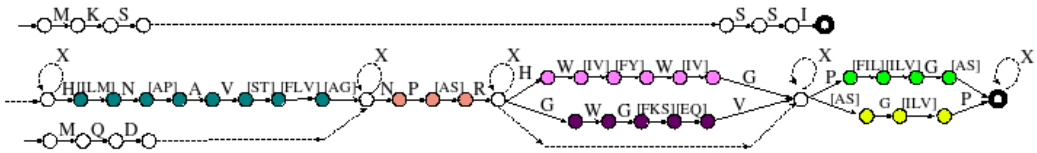


FIG. 4.10 – Automate résultat, exhibant les zones caractéristiques, les chemins exceptions à ces zones, ainsi que les zones de gaps reliant les zones caractéristiques.

4.6 Propriétés physico-chimiques

Malgré tous ces efforts de modélisation dans le cadre de la théorie des langages, il ne faut pas perdre de vue que les acides aminés ne sont pas réduits à de simples lettres parmi un alphabet de taille 20. En effet, le chapitre 1 nous a appris à connaître leur nature chimique ainsi que les propriétés physico-chimiques pour lesquelles on pouvait former des groupes.
Devant des jeux de données correspondant à des familles très restreintes car trop peu de membres sont séquencés, il n'est pas possible d'observer toutes les variations possibles d'une position en acides aminés. C'est pourquoi nous avons souhaité apporter un outil permettant d'étendre un petit ensemble d'acides aminés alignés au groupe physico-chimique qu'il semble remplir le mieux.
Nous savons que les matrices de substitutions sont construites à partir de nombreux jeux similaires parfaitement alignés. Chaque colonne de l'alignement permet d'observer des fréquences de variations pour chaque acide aminé. Nous proposons ici d'utiliser ce même concept afin de vérifier si la répartition des acides aminés permet d'identifier une propriété physico-chimique. Le but étant de substituer certaines transitions présentant plusieurs acides aminés dans un automate, par des transitions étendues aux acides aminés du groupe physico-chimique identifié.
L'approche prend en entrée un jeu $\mathcal{G}$ de groupes physico-chimiques d'acides aminés. Les groupes se recouvrent potentiellement (voir au chapitre 1 le diagramme de Venn proposé pour les groupes de Taylor).
De façon naïve il suffirait de tester pour chaque ensemble P d'acides aminés alignés, s'il est équivalent à un des groupes pris en compte. Mais cette approche ne peut être appliquée qu'aux petits groupes (Nevill-Manning et coll., 1998). Cependant il faudra un grand nombre de séquences dans l'échantillon d'apprentissage afin d'identifier de façon identique des groupes plus importants (il suffit d'imaginer par exemple la probabilité d'aligner 13 acides aminés du groupe des hydrophobes dans un petit jeu de protéines homologues).
Nous proposons donc d'utiliser un test statistique afin de décider si le multi-ensemble P a été généré à partir d'un groupe physico-chimique G ou non (voir algorithme 4.6).
Soit deux états q_1, q_2 d'un automate, et P l'ensemble de tous les acides aminés permettant d'atteindre q_2 depuis q_1, et soit n le nombre total de séquences utilisant ces transitions. Nous décidons de remplacer cet ensemble d'acides aminés P par le plus

petit groupe physico-chimique G incluant P en fonction d'un *likelihood ratio test* (test de vraisemblance).
Pour calculer ce ratio, nous utilisons la probabilité de fond p_a de chaque acide aminé a et nous estimons la probabilité $p_{a|G}$ pour cet acide aminé d'appartenir au groupe G par la probabilité $p_{a|G} = c_G p_a$, telle que c_G est proportionnel au facteur de redistribution d'acides aminés manquant : $c_G = \frac{1}{\sum_{a \in G} p_a}$.
Dans ces conditions, nous pouvons comparer la vraisemblance L_G de G lorsque n acides aminés sont produits depuis G, avec la vraisemblance lorsque les acides aminés sont produits depuis P par le ratio : $LR_{G/P} = \frac{L_G}{L_P} = \left(\frac{\sum_{a \in P} p_a}{\sum_{a \in G} p_a}\right)^n$. Etant donné un seuil λ_G, nous testons l'extension de P vers G et l'écartons lorsque $LR_{G/P} < \lambda_G$. Si l'extension G est écartée, il n'y a aucune évidence de la propriété d'un groupe physico-chimique dans le jeu des acides aminés P. Dans un tel cas, il est possible d'évaluer si P a été généré de manière aléatoire et ainsi le remplacer entièrement par l'alphabet Σ, ou bien laisser tel quel un groupe dont la composition est suffisamment importante pour être conservée.
En remplaçant G par Σ et en introduisant un seuil λ_Σ, nous testons de manière identique l'extension de P vers Σ en l'écartant si $LR_{\Sigma/P} = \frac{L_G}{L_P} = (\sum_{a \in S} p_a)^n < \lambda_\Sigma$.

Algorithme 8 Identification et extension aux propriétés physico-chimiques

Entrées Automate A, jeu de groupes d'acides aminés $\mathcal{G}$, seuils $\lambda_G, \lambda_\Sigma$.

```
Pour chaque (q1, q2) ∈ A faire
  P ← {a ∈ Σ | (q1, a, q2) ∈ A}
  Si P ≠ ∅ alors
    G ← LE_PLUS_PETIT_GROUPE{G ∈ 𝒢 | P ⊆ G}
    Si LR_{G/P} ≥ λ_G alors
      Pour chaque a ∈ G \ P faire
        AJOUT{A, (q1, a, q2)}
      Fin pour
    Sinon si LR_{Σ/P} ≥ λ_Σ alors
      Pour chaque a ∈ Σ \ P faire
        AJOUT{A, (q1, a, q2)}
      Fin pour
    Fin si
  Fin si
Fin pour

return A
```

4.7 Algorithme complet de l'approche

En reprenant l'algorithme 4.4 de l'approche globale, et en y ajoutant les concepts d'exception (voir 4.5) et de propriétés de groupes physico-chimique (voir 4.6), nous pou-

vons à présent décrire l'algorithme complet (voir algorithme 9) de l'approche. Nous introduisons une fonction $IdentificationZonesCaracteristiquesEtNonCaracteristiques(A, Q)$ qui permet simplement à partir d'un automate et d'un quorum de lister les zones caractéristiques et les zones non-caractéristiques selon la définition donnée en 4.4.

Algorithme 9 Approche complète incluant la fusion du PLMA et les zones de Gap, les Exceptions, l'identification des groupes physico-chimiques ainsi que l'extension aux groupes physico-chimiques.

Entrées un jeu de séquence S, une fonction de partition $ConstruitPLMA()$, un quorum Q, jeu de groupes d'acides aminés $\mathcal{G}$, seuils $\lambda_G, \lambda_\Sigma$
Sorties un modèle de type automate A

```
P ← ConstruitPLMA(S)
A ← CreerMCA(S)
A ← FusionPositions(A, P)
LZ ← IdentificationZonesCaracteristiquesEtNonCaracteristiques(A, Q)
Pour chaque Z ∈ LZ faire
  Si Z ≠ Caracteristique alors
    Si Z ≠ Exception alors
      A ← FusionGap(A, Z)
    Fin si
  Fin si
Fin pour
ExtensionPhysicoChimique(A, G, λ_G, λ_Σ)
Retourner(A).
```

4.8 Conclusion générale de l'approche

Nous avons donc proposé une démarche qui consiste à produire un alignement permettant d'identifier des zones similaires de manière locale et partielle, puis à utiliser cet alignement comme base des fusions dans un automate non-déterministe représentant un échantillon de séquence d'une famille de protéine.
Cette approche a connu de nombreux ajustement concernant les objets étudiés (fragments, SFP, Bloc de PLMA, PLMA), les scores affectés à ces objets, ainsi que les contraintes entraînées par la combinaison de ces objets.
Les méthodes d'apprentissage par fusion d'états, étendues à la fusion de fragments ou d'ensembles de fragments, semblent capables de produire des automates simples et pratiques pour l'analyse des informations contenues dans les séquences.
Les concepts d'Exceptions ou encore d'extension aux groupes physico-chimiques ont été introduits suite aux premières expériences et aux différents constats de la réalité des échantillons de séquences.
Puisque l'on a désormais une approche complète, nous allons pouvoir étudier ses performances au travers des principales expérimentations menées lors de cette thèse.

Troisième partie

Evaluation

Nous avons donc les moyens de proposer aux biologistes de générer des automates à partir de leurs familles biologiques d'intérêt. Le niveau d'expressivité choisi est novateur. Cependant il est nécessaire d'évaluer la qualité des modèles obtenus. Le but est de pouvoir comparer les automates à d'autres modèles d'une part. Et d'autre part, vu que nous pouvons obtenir plusieurs automates en faisant varier le quorum, il est important d'avoir des mesures capables d'indiquer celui qui possède les meilleurs propriétés recherchées.

Chapitre 5

Implémentation

5.1 Codage

Le codage de la dernière implémentation du programme Protomata-Learner a été fait en Python. Il a nécessité plusieurs librairies dont Gabios-Lib (Abdeddaïm et Morgenstern, 2000) permettant d'évaluer certaines contraintes.
Une partie indépendante du programme est consacrée à la caractérisation et à la découverte d'un PLMA pour un jeu de séquence donnée.
La deuxième partie est consacrée à la construction d'automates dans plusieurs formats dont certains visibles grâce à l'outil Dot de la série de logiciel GraphViz. Cette partie est également le siège de l'implémentation de l'identification des différentes zones, dont celles impliquant les chemins exceptions dont une partie développée dans le cadre d'un stage au sein de l'équipe (Hoang, 2006).

Le programme est a fait l'objet d'un dépôt APP. Nous avons également choisi d'en faciliter l'accès par la création d'une interface web dans le cadre de notre collaboration avec la plateforme de la Génopole Ouest.
Cette interface réalisée en PHP permet aux biologistes de proposer leur jeux de séquences en ayant accès à tous les paramètres de notre approche. Ils se voient retourner des sorties d'images représentant les automates qu'ils peuvent analyser visuellement ou choisir d'y faire parser des banques pour sortir d'éventuels nouveaux candidats.
Nous avons également procédé à l'installation et à la configuration de l'outil sur le cluster de la Génopole Ouest sur lequel les travaux lancés par l'interface web sont exécutés.

5.2 Acceptation d'une séquence dans un modèle

Nous avons vu au chapitre 2 qu'un automate acceptait une séquence faisant ainsi partie de son langage lorsqu'il existait un chemin d'un état initial à un état final dont les valeurs des transitions correspondaient à des successions de lettres dans la séquence. Pour ce qui est des motifs, il existe différentes méthodes de recherche de motifs dans des banques de séquences dont les meilleures stratégies consistent à prétraiter par indexation les banques. Cependant, les acceptations de façon discrète (sans erreur) limitent

réellement les capacités de reconnaissance des motifs. C'est pourquoi il est pratique d'introduire la possibilité de scanner des banques avec erreur. Il s'agit de définir un seuil à partir duquel on considérera une séquence comme étant acceptée par le modèle, même si pour cela il a fallu y faire plusieurs substitutions.
Un algorithme classique venant de la programmation dynamique permet ce type de scans. C'est l'algorithme de Viterbi. Une implémentation pour les motifs et automates, adaptée à la recherche de séquences biologiques, en a été faite par le programme Wapam.
Sur cette même idée nous avions développé notre propre approche en utilisant une matrice de coût de substitution. Basée sur la matrice de substitution Blosum62, notre matrice permettait d'avoir une première évaluation de la distance d'une séquence par rapport à un automate, ou plutôt du coût en terme de sommes de coûts de substitution afin que la séquence soit acceptée dans l'automate.
Ne pouvant nous satisfaire de ce parsing en *distance*, par la suite nous avons produit un autre type de score qui prend en compte les pondérations des différents acides aminés présents par transitions. Nous avons alors profité d'un stage (Mahé, 2007) qui nous a permis de convertir nos automates de protéines au format des automates WA demandés par Wapam, puis dans sa nouvelle version nommée Wascan. Nous faisons référence à cette méthode d'évaluation de la proximité d'une séquence avec un modèle automate par le terme de parsing *poids*, les résultats étant des scores semblables aux résultats de type Blast. Plus le score obtenu par une séquence dans un automate est élevé, plus la séquence est proche de la famille. A l'inverse, les faibles scores, voire les scores négatifs indiquent une séquence particulièrement différente de la famille.

5.3 Evaluer le modèle avec une mesure *Minimum Description Lenght* (MDL)

En l'absence d'indications sur les performances en reconnaissance, nous avons souhaité introduire une mesure issue de la théorie de l'information et de notions telles que l'entropie, définie par Shannon. En effet, Protomata-Learner ne génère pas que de simples outils de discrimination, c'est aussi une nouvelle forme de modèles permettant d'observer la structure du jeu de séquences. La mesure choisie pour évaluer ce travail de modélisation est une mesure de type "Minimum Description Lengh (MDL)". La mesure MDL est une formalisation du principe du rasoir d'Occam dans laquelle la meilleure hypothèse pour un ensemble de données est celle qui conduit à la plus grande compression des données. MDL a été présenté par Jorma Rissanen (Rissanen, 1978).
Pour la définition d'une mesure de type MDL appliquée à notre approche, nous considérons que :

- La donnée à compresser est le jeu de séquence S représentant la famille protéique
- Le modèle M permettant la compression est un automate A généré par Protomata-Learner
- La mesure MDL (en bits) sera égale au codage (en bits) du modèle M ajouté au codage (en bits) du jeu S sachant le modèle M

– Parmi tous les automates générés en faisant varier un paramètre comme le quorum, nous chercherons celui qui minimise la mesure MDL

Nous allons donc voir le choix de codage (en bits) qui a été fait pour chaque élément du calcul MDL.

Nombre de bits nécessaires pour coder une des lettres représentant les acides aminés :

Soit Σ l'ensemble des Acides Aminés

Notons $\sharp x$ le nombre de bits nécessaires pour coder x.

On a alors :

$$a \in \Sigma : \sharp a = \log_2 |\Sigma|$$

Codage de l'ensemble des Séquences : S

Notons $S = \langle s_1, \ldots, s_{|S|} \rangle$ l'ensemble des séquences dont le codage est :

$$\sharp(S) = \sum_{i=1}^{|S|} \sum_{j=1}^{|S_i|} \log_2 |\Sigma|$$

Codage du modèle : M

Soit un automate $A = \langle \Sigma, Q, E, Q_I, Q_F \rangle$

Sachant que l'on étiquette chaque transition par un acide aminé ou un groupe d'acides aminés, le codage de l'ensemble sans doublons des groupes d'acides aminés G nécessite alors :

$$\sharp(G) = \sum_{g \in G} |g| \times \log_2 |\Sigma|$$

L'étiquetage de chaque transition t nécessite alors :

$$\sharp(t) = \log_2 |\Sigma + G|$$

Le codage de l'ensemble des transitions E (étiquetage et états participant à chaque transition) de l'automate devient donc :

$$\sharp(E) = |E|.\sharp(t).2 \log_2 |Q|$$

Le codage du modèle M représenté par l'automate A nécessite alors :

$$\sharp(M) = \sharp(G) + \sharp(E)$$

Codage d'un ensemble de séquences sachant le modèle : S/M

Sachant les positions associées à une séquence s_i dans l'automate, nous allons estimer le coût du codage d'un parcours permettant de retrouver la séquence à partir de l'automate.

Le codage du choix C_p d'un acide aminé d'une position p d'une séquence s_i sachant l'ensemble e_t d'acide aminés proposé par l'étiquetage d'une transition t est le suivant :

$$\sharp(C_p) = \log_2 |e_t|$$

À partir de la transition t parcourue par la séquence s à la position p, le codage du choix T_p de la transition $t+1$ parcourue par la séquence s à la position $p+1$, parmi l'ensemble des K transitions positions possibles après t est :

$$\sharp(T_p) = \log_2 |K|$$

Pour chaque position p d'une séquence s_i, sachant le modèle M, le codage est de :

$$\sharp(p/M) = \sharp(C_p) + \sharp(T_p)$$

D'où le coût du codage d'une séquence s_i, sachant le modèle M :

$$\sharp(s/M) = \sum_{p=1}^{|s|} \sharp(p/M)$$

Et enfin le codage de l'ensemble des séquences S sachant le modèle M :

$$\sharp(S/M) = \sum_{s \in S} \sharp(s/M)$$

Codage MDL total :

$$\sharp(MDL) = \sharp(M) + \sharp(S/M)$$

5.4 Etude de cas sur le clan Pfam de la super-famille des Viral Nucleic Acid Binding

La base de données de HMM profils Pfam (Bateman et coll., 2004) a été introduite au chapitre 2. Nous savons que chaque entrée correspond à un profil de type HMM représentant une famille de protéines. Pfam a la particularité de posséder des entrées nommées *clans*. Ces clans sont en fait des ensembles regroupant plusieurs familles Pfam en relation de par les similarités de séquences ou de structures de leur membres.
Lorsque l'on regarde les séquences regroupées sous le nom de clan, on s'aperçoit qu'il existe des domaines partagés par toutes les séquences et d'autres par quelques Pfam du clans. On comprend alors les limites des HMM profils qui sont incapables de représenter

des subdivisions en sous-familles sur le même modèle.
Afin d'étudier la structure de notre modèle, nous allons étudier ce que l'on obtient lors d'un apprentissage sur un clan Pfam. Les captures d'écran ont été produite à partir de l'interface web de la plateforme de bioinformatique de la Génopole-Ouest.

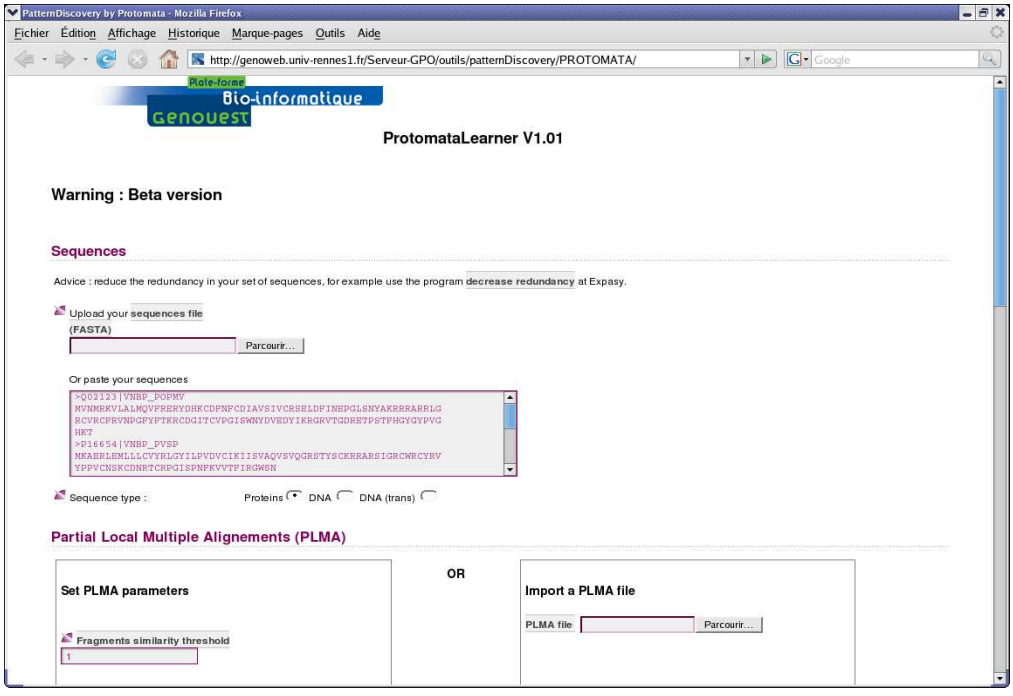


FIG. 5.1 – On utilise un jeu de séquences d'apprentissage (ici le clan PFAM connu sous le nom de "Viral nucleic acid binding").

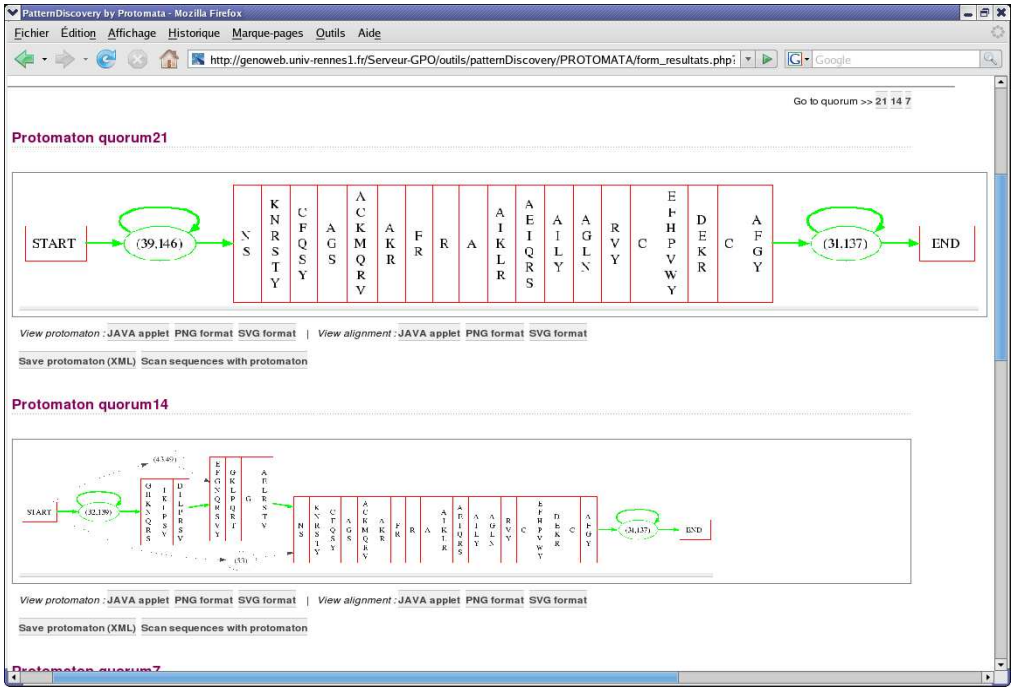


FIG. 5.2 – On obtient plusieurs automates en fonction de différents quorum

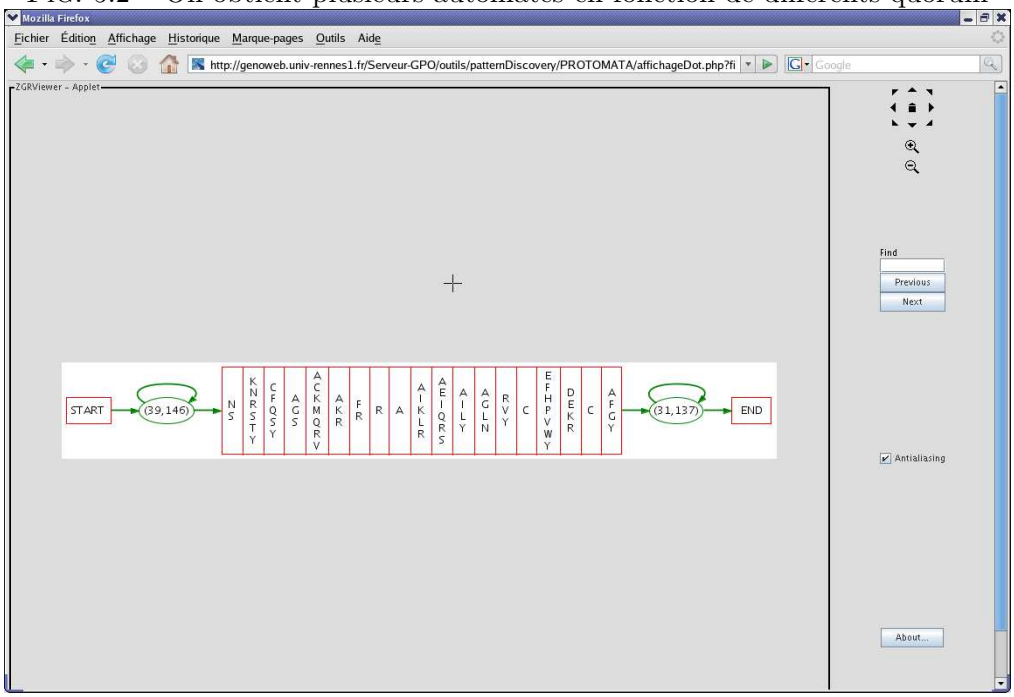


FIG. 5.3 – Pour un quorum de 100%, nous avons un automate équivalent à un motif de type Prosite.

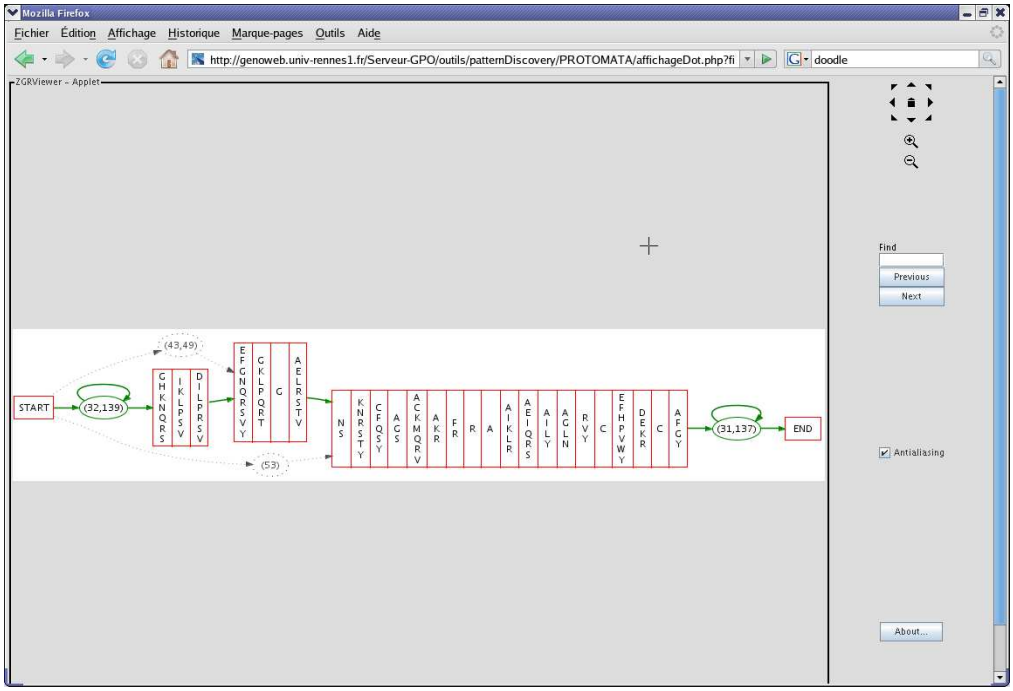


FIG. 5.4 – En descendant en quorum, nous découvrons des blocs contenant moins de séquences, ainsi que des exceptions à ces blocs.

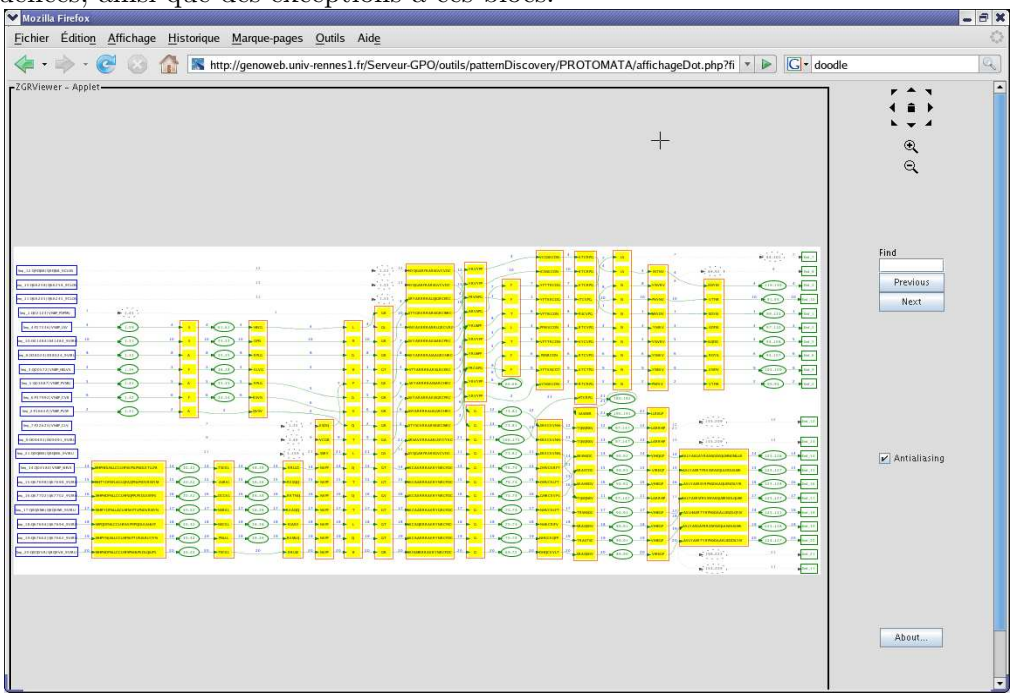


FIG. 5.5 – Enfin, la vue “plma” nous permet, pour des quorums encore plus petits, d’observer le comportement des familles de séquences.

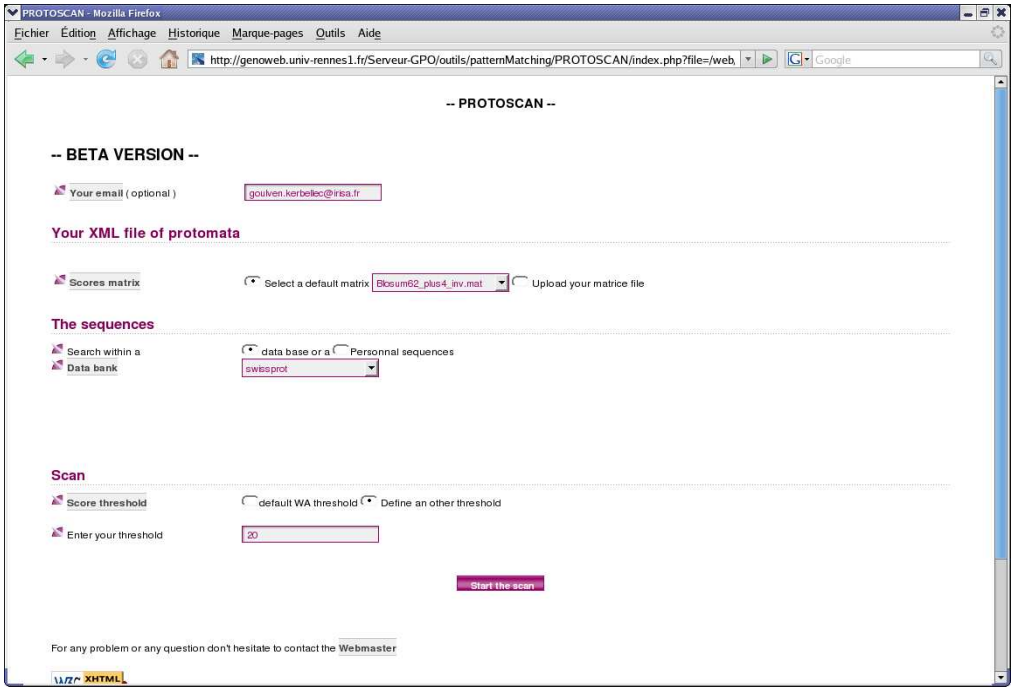


FIG. 5.6 – Pour chaque automate ou tout simplement pour celui qui montre le meilleur score MDL, nous pouvons scanner cet automate sur une base de donnée.

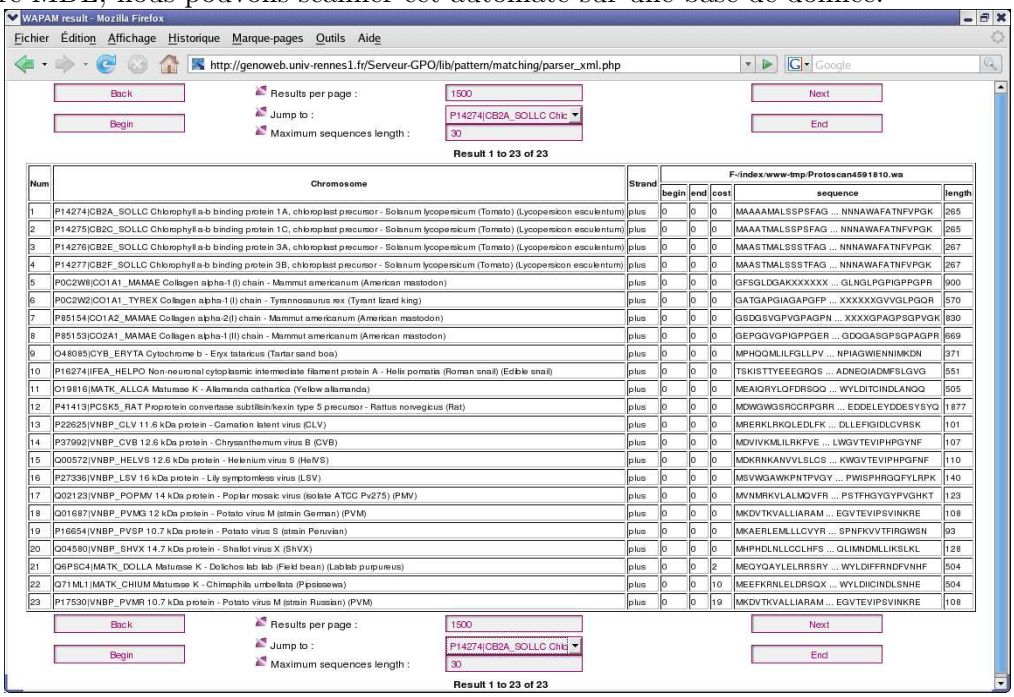


FIG. 5.7 – Voici le résultat d'un scan, avec les séquences valides et leur score de parsing.

Chapitre 6

Expériences d'Apprentissage de Protomates

Suite à plusieurs implémentations de notre approche, nous avons choisi de nommer *Protomates* les modèles de type automate que nous obtenons suite à apprentissage sur un jeu de protéines. Au cours des différentes distributions, expériences et publications, le nom de notre programme à convergé vers celui de *Protomata-Learner*.
Ce chapitre présente les expériences menées autour des applications permises par les différentes versions de *Protomata-Learner*.
Cette thèse a une application particulièrement ciblée sur la famille de protéines dite des TNF, impliquées dans des maladies cancéreuses. C'est pourquoi, nous avons produit de nombreux résultats sur cette famille qui est apparue comme particulièrement difficile pour les approches classiques.
Cependant nous souhaitions également avoir l'approche la plus générale possible de manière à ce que nous puissions traiter chaque famille de protéine que l'on nous propose. De plus, il était important de pouvoir valider l'approche sur des données que nous connaissions. Ainsi nous avons profité de la collaboration avec C. Delamarche (UMR CNRS 6026) pour faire les premiers essais sur les protéines MIP. Puis nous avons élargi les cibles et évalué le comportement de l'approche sur les TNF.

6.1 Validation de l'apprentissage

Le chapitre 3 nous a permis de suivre le processus permettant l'apprentissage d'un automate à partir d'un ensemble de séquences appartenant à une famille de protéines. Il s'agit désormais d'étudier les différentes techniques d'évaluation de la qualité de ces automates.
Tout d'abord, nous allons détailler la technique la plus classique, qui consiste à évaluer les capacités de discrimination des modèles. C'est-à-dire la capacité pour le modèle appris, à reconnaître de nouveaux membres de la famille d'intérêt et cela sans se tromper. Pour cela, nous avons besoin de plusieurs jeux de séquences. L'apprentissage nécessite un jeu d'apprentissage positif ne contenant que des séquences appartenant à la famille,

et éventuellement un jeu d'apprentissage négatif. Quant à la validation, elle se fait avec un jeu de validation positif et un jeu de validation négatif.
Après avoir inféré le modèle sur le(s) jeu(x) d'apprentissage, nous utiliserons ce modèle pour scanner les jeux de validations. Il en résultera des séquences acceptées par le modèle et des séquences rejetées par le modèle. Nous introduisons alors les notations qui nous permettent d'évaluer le succès de la discrimination.

Définition 6.1 *Un vrai positif (VP) est une séquence du jeu de validation positif qui est acceptée par le modèle appris.*

Définition 6.2 *Un faux positif (FP) est une séquence du jeu de validation négatif qui est acceptée par le modèle appris.*

Définition 6.3 *Un faux négatif (FN) est une séquence du jeu de validation positif qui est rejetée par le modèle appris.*

Définition 6.4 *Un vrai négatif (VN) est une séquence du jeu de validation négatif qui est rejetée par le modèle appris.*

Les trois mesures qui rendent compte de la capacité du modèle à discriminer sont le rappel, la précision et la F-mesure. Le rappel montre la capacité à reconnaître des membres de la famille. La précision montre la capacité à ne pas se tromper en choisissant de mauvais candidat. Lorsque l'on obtient un bon rappel et une mauvaise précision, on peut dire que l'on a surgénéralisé lors de l'apprentissage. Lorsque l'on obtient une bonne précision et un mauvais rappel, on peut dire que l'on a surspécifié lors de l'apprentissage. La F-mesure est une mesure qui rend compte de la qualité du compromis entre les deux autres mesures.

Définition 6.5 $Rappel = \frac{\sharp(VP)}{\sharp(VP)+\sharp(FN)}$

Définition 6.6 $Precision = \frac{\sharp(VN)}{\sharp(VN)+\sharp(FP)}$

Définition 6.7 $F - mesure = \frac{2.Rappel.Precision}{Rappel+Precision}$

6.2 Leave-One-Out

Dans les expérimentations, nous avons donc évalué à plusieurs reprises des modèles inférés sur des jeux d'apprentissage et testés sur des jeux de validation.
Si l'on disposait de plusieurs centaines de séquences par famille il suffirait d'en extraire une fraction (1/10 ou 1/3 par exemple) et d'en faire notre jeu de validation positif, tandis que l'apprentissage serait fait sur les séquences restantes. Dans un cas plus difficile, avec une centaine de séquences par famille, l'idée serait de compenser le manque d'information par une réitération du choix du jeu de séquences de validation positif. On aurait alors, par exemple, 10 expériences testant à chaque fois un nouveau modèle

appris sur 9/10 des séquences, et évalué sur les 1/10 restant. Puis nous pourrions alors prendre la moyenne des résultats obtenus à la suite de ces évaluations. Ceci s'appelle la validation croisée.
Cependant, nous n'avons pas toutes les séquences existant dans la nature et potentiellement membres d'une famille. Pire, il est certain que, compte-tenus de la longueur des séquences et du nombre de variations possibles, la nature n'a pas exploré toutes les solutions possibles. C'est donc avec un manque cruel d'information en quantité de séquences que nous avons du travailler. Sur des familles de quelques dizaines de membres, la technique de validation choisie est une validation croisée impliquant une seule séquence de validation à chaque itération. Elle est nommée le Leave-One-Out.
Notons que le manque de séquences couvrant l'ensemble des possibilités pour une fonction est le cas le plus courant. De plus, le jeu de contre-exemple n'est pas beaucoup plus évident à établir car il n'existe pas forcément beaucoup de séquences n'appartenant pas à la famille étudiée, mais suffisamment similaires aux membres de cette famille.

6.3 Familles d'intérêt

6.3.1 Construction de jeux de données

En bioinformatique, il est arrivé que certaines validations d'outils se fassent sur des jeux de données générés de façon aléatoire. Nous n'avons pas souhaité utiliser ce qui pour notre problématique nous semble être une validation qui donne des indications théoriques mais qui peut se révéler inefficace dans les majorité des cas pratiques.
Nous allons voir dans les exemples de familles choisies, la difficulté à bien définir le jeu qui représentera la famille, en choisissant les séquences qui le composent. Supposons que nous souhaitions étudier une famille λ. A priori il suffirait de choisir dans une banque de données tous les membres annotés comme appartenant à la famille. Mais voici les biais qui peuvent entraîner la construction d'un jeu non représentatif :

- le choix de la banque de données (le bon compromis entre une banque annotée à la main mais ayant peu d'entrées, et une banque avec de nombreuses entrées générées par traduction de l'ADN)
- la présence d'une séquences dans plusieurs espèces
- les séquences ayant leurs fonctions annotées "hypothétiques" ou "par similarité"
- une définition trop floue des propriétés de la famille

En effet, un jeu de séquences sera beaucoup plus fiable si on le choisit dans Swissprot que dans Trembl par exemple. Cependant la qualité de l'annotation est inversement proportionnelle à la taille du jeu de séquences.
D'autre part, il faut faire attention aux séquences surreprésentées pour des raisons historiques. Plus une protéine a été séquencée tôt chez l'homme par exemple, plus on aura eu le temps de trouver des séquences similaires provenant de gènes orthologues dans des espèces plus exotiques.

6.3.2 *Major Instrinsic Proteins* (MIP)

Description

Les protéines de la famille MIP (Major Intrinsic Proteins) sont impliquées dans les mécanismes de maintien de l'homéostasie cellulaire (Agre et Kozono, 2003).
La famille MIP désigne des canaux qui sont impliqués dans le transport de l'eau et/ou de petits solutés non chargés (principalement le glycérol) à travers les membranes biologiques. Pour faciliter la description des MIP, il est habituel de distinguer deux types fonctionnels : Les aquaporines et les facilitateurs du glycérol. La découverte des aquaporines par Peter AGRE (prix Nobel de chimie 2003) a révolutionné les connaissances biophysiques sur les mécanismes de perméabilité des membranes biologiques. L'analyse de plusieurs pathologies humaines confirme que les aquaporines sont fondamentales dans l'organisme : dysfonctionnement rénal, oedème cérébral et cardiaque, vision, intoxication.
Les aquaporines ont été identifiées dans les trois règnes du vivant. Elles permettent un transport passif rapide et sélectif de l'eau au travers de la membrane plasmique des cellules tout en excluant le passage des ions. Le dysfonctionnement d'aquaporines est associé à un nombre de plus en plus grand de pathologies humaines : Diabète insipide, cataracte, oedèmes. La protéine modèle des aquaporines est AQP1 chez les mammifères et AqpZ chez *E. coli*.
Les facilitateurs du glycérol permettent le transport facilité du glycérol et/ou de petits solutés non chargés au travers de la membrane plasmique tout en excluant l'eau et les ions. Le modèle historique de ce groupe fonctionnel est la protéine GlpF de E. coli, qui forme dans la membrane un canal de diffusion facilitée au glycérol. Après phosphorylation dans le cytoplasme, le glycérol ne peut plus franchir le canal en sens inverse.
Nous connaissons 13 gènes de protéines MIP chez l'homme et 38 gènes chez la plante *Arabidopsis thaliana*. Les membres de la famille MIP possèdent environ 300 acides aminés et présentent une organisation structurale commune comportant 6 alpha-hélices entourant un pore central. Cette organisation correspond au modèle topologique proposé par P. Agre et ses collaborateurs en 1994 et connu sous le nom de "modèle en sablier". Le modèle en sablier à été validé au plan structural sur l'aquaporine AQP1 et sur le facilitateur du glycérol GlpF. Toutes les protéines MIP présentent de nombreux résidus hautement conservés localisés le long de la séquence. On remarque en particulier, une bonne conservation du motif asparagine-proline-alanine (NPA), présent dans les 2 petites hélices HB et HE qui plongent symétriquement à l'intérieur du pore et qui participent à la zone de constriction du canal.
La structure des 2 protéines modèles de la famille MIP, AQP1 et GlpF, a été déterminée par cristallographie aux rayons X avec une résolution de 2,2 Å (Sui et coll., 2001; Fu et coll., 2000). Ces études montrent que AQP1 et GlpF cristallisent sous forme d'homotétramères dans lesquels chaque monomère forme un « canal actif ». Ces structures ont permis de proposer un mécanisme moléculaire de fonctionnement des canaux MIP, dans lequel la sélectivité à l'eau ou au glycérol s'expliquent par la distribution d'acides aminés spécifiques tout le long du pore et au niveau de la zone de restriction (Fujiyoshi et coll., 2002; Stroud et coll., 2003) . Des expériences de mutagenèse dirigée confirment

le rôle clé de certains acides aminés localisés à l'intérieur du pore. Cependant, il faut noter que ce modèle n'explique pas totalement les mécanismes moléculaires de sélectivité des substrats in vivo, par exemple le fonctionnement des canaux mixtes (eau / glycérol), la régulation bidirectionnelle du flux d'eau ou de glycérol et l'imperméabilité aux ions.

Intérêt pour l'approche

Les protéines MIP nous ont semblé être un jeu très intéressant pour nos premières expérimentations. En effet, le fort taux de similarité entre les séquences permet de situer rapidement les domaines connus pour cette super-famille et donc de vérifier si nos modèles présentent bien ces domaines. De plus, l'existence de sous-familles (aquaporines, facilitateurs du glycérol) fonctionnelles nous permet d'évaluer le pouvoir de discrimination des différentes approches de découverte de motifs.

Jeux de séquences choisis pour les expériences sur les MIP

Au moment de la préparation des données, la base UNIPROT contenait un ensemble U de 911 protéines annotées comme étant des membres de la famille MIP. Comme expliqué au paragraphe 6.3.1, la qualité de l'apprentissage et de son évaluation nécessitent que l'on écarte certaines séquences. Ce travail a été effectué en collaboration avec C. Delamarche, professeur en biologie, travaillant sur cette famille.
A partir de U, nous avons extrait un jeu nommé T composé des 159 séquences de protéines dont l'annotation est considérée comme fiable puisque provenant de la base de référence Swissprot.
De cet ensemble T, a été identifié un jeu A ne contenant que 79 séquences ayant été annotées suite à une réelle expérience biologique, les autres étant annotées par similarité avec les premières.
Puis nous avons dégagé certaines séquences de A de manière à posséder un jeu M dans lequel on ne peut trouver deux séquences partageant une identité supérieure à 90%. La taille de M est de 44 séquences. C'est notre jeu étalon pour les MIP. De cet ensemble, C. Delamarche a identifié 24 séquences spécifiques à l'eau (jeu $W+$) et 16 séquences non-spécifiques à l'eau (jeu $W-$) puisque pouvant laisser transiter du glycérol ou de petits solutés. Nous avons créé également un jeu de séquences contrôle C composé de séquences proches des séquences MIP (en utilisant les premiers hits du programme Blast) et identifiées par les experts comme étant en dehors de la famille.

6.3.3 *Tumor Necrosis Factor* (TNF)

Description

Le développement exponentiel des cellules cancéreuses témoigne de l'absence de leur apoptose. En effet, les processus conduisant à l'autodestruction ou suicide de la

cellule sont affectés. Les traitements actuels, principalement chirurgie, chimiothérapie et radiothérapie, rencontrent le problème de résistance de certaines cellules cancéreuses. Les progrès de la génomique liés au projet "génome humain" tentent de pallier à ces problèmes en ouvrant de nouvelles voies. Ainsi, les différentes protéines responsables de signaux induisant l'apoptose ont été découvertes et classées. Nous pouvons distinguer la super-famille des TNF (Ashkenazi, 2002) qui est associée aux différentes familles de ligands (environ 18), et celle des TNFR associée aux familles des récepteurs (environ 28 "receptors") de ces ligands. Le nom de TNF est aussi celui d'une sous-famille de la super-famille des TNF, la raison historique en est que la sous-famille TNF a été découverte en premier (1975).
Nous disposons de nombreuses familles fonctionnelles dont les combinaisons ligand / récepteur sont importantes. Leur caractérisation passe donc par l'identification des aspects structurels comme l'implantation transmembranaire des récepteurs, mais aussi par des sites de liaisons entre ligands et récepteurs. Ainsi, au sein de la super-famille des TNFR, les régions extracellulaires N-terminales sont les plus conservées (environ 65%). Elles représentent de 1 à 6 occurrences (généralement 3) de CRD (Cysteine-Rich Domains) se présentant dans le milieu extracellulaire sous la forme d'hélices alpha stabilisées par des liaisons de type "pont disulfure". Dans la super-famille des TNF, la région C-terminale est sujette à diffusion dans le milieu extracellulaire et est la plus conservée (environ 30%).
La fonction principale des TNF est de changer la conformation du récepteur pour déclencher le signal. Par exemple, les ligands FASL se lient et transforment la conformation de récepteurs de la famille FAS. Ces récepteurs ont pour fonction de transmettre le signal d'apoptose vers l'intérieur de la cellule. C'est pourquoi ils sont signés d'un domaine intracellulaire, ou adaptateur, nommé FADD (FAS-associated death domain) dont le rôle est d'activer les caspases (enzymes initiant l'apoptose). Cependant, d'autres sous-familles des TNFR utilisent des adaptateurs différents tels que les TRADD (TNFR-associated death domain) qui contrôlent, par l'intermédiaire de kinases, la cascade de phosphorylation des gènes du système immunitaire. Les TRADD peuvent aussi intervenir indirectement par l'intermédiaire des FADD.
Il existe dans certaines familles des cas particuliers ou dégénérés qui se traduisent par l'absence d'expression de la fonction alors que la similarité avec les autres membres de la famille est importante. Ce qui pourrait sembler anecdotique pour d'autres super-familles est ici d'une importance capitale. En effet, les Decoy receptors (DcRs) font partie des TNFR et ont la particularité de se lier à des TNF sans pour autant déclencher de signal. Les conséquences peuvent être dramatiques, car dans le cas d'une surexpression, les DcRs entrent en compétition avec les autres TNFR et empêchent alors l'apoptose de s'initier. Ainsi, on peut signaler que les DcR1 et DcR2 entrent en compétition avec les DR4 et les DR5 (récepteurs de APO2L/TRAIL). De même, les ligands FASL peuvent être bloqués par des DcR3 au lieu des FAS.

Intérêt pour l'approche

L'intérêt pour la famille des TNF est porté par le fait qu'il existe probablement des ligands non-connus et capables de se liéer à des récepteurs qui ont été identifiés précédemment. Cette famille est aussi intéressante par les difficultés qu'elle semble présenter. En effet, les séquences sont assez peu similaires entre elle. L'essentiel de leur structure secondaire est constitué de feuillets Béta qui sont généralement bien moins conservés en séquence que des hélices alpha. Nous verrons également par la suite que le motif de référence stocké dans la base de données Prosite ne permet pas une discrimination parfaite.

Jeux de séquences choisis pour les expériences sur les TNFs

Nous nous intéressons donc principalement à la famille des ligands des TNF. Certaines séquences sont connues sous des formes assez proches dans différentes espèces. Pour ne pas biaiser l'apprentissage, nous nous intéressons ici uniquement aux ligands connus des TNF de l'espèce humaine. Soit 18 séquences formant le jeu positif $N+$ au moment de la création du jeu. Le pourcentage moyen d'identité y est de 33,6% avec un minimum approchant de 0% et un maximum de 70%.
Nous avons également construit un jeu de négatifs $N-$. Ce jeu contient d'une part 4 séquences apparaissant comme faux positif du motif Prosite des TNF, et d'autre part 16 séquences appartenant à la super famille des cytokines mais n'étant pas des TNF.
Le pourcentage moyen d'identité entre le jeu positif et le jeu négatif est 28,56% avec un minimum de 0% et un maximum de 80%.

6.4 Expérience d'apprentissage d'un motif MIP

Nos premières implémentations et expériences ont été produites à partir de la version Strictement SFP (voir algorithme en 3.14) de notre approche.Les concepts de PLMA et Bloc de PLMA (3.8) n'ont été clairement posés que par la suite. Cependant, l'idée d'assembler des fragments significativement similaires de façon itérative (paire après paire) devait nous conduire, par transitivité, à l'obtention des zones similaires connues dans les familles de protéines.
Nous avons vu (algorithme 3.14) que l'ordonnancement des paires de fragments était variable selon la mesure avec laquelle on choisissait de les comparer. Nous avons alors exploré trois types de scores différents (voir score Dialign, Support et Implication, décrits dans le chapitre 3). Chaque score permet ainsi d'orienter la construction des automates vers des propriétés particulières.
Par la suite, nous ferons référence à plusieurs paramètres utilisés lors de l'exécution de l'approche. Certains sont tout simplement liés au fait que nous utilisions une option du programme Dialign2 (Morgenstern, 1999) pour générer des paires de fragments. L'option de pré-traitement "-afc" permet une évaluation d'un score de similarité entre deux fragments que l'on associe à une valeur seuil (seuil Dialign dont l'option est accessible par "-thr") pour produire une liste de SFP utilisable par nos algorithmes.

TAB. 6.1 – Comparaison de 4 motifs de MIP.

Méthode	Precision	Rappel	F-mes.	Motif
Prosite (reference)	95%	91%	0.93	[HNQA]DNP[STA][LIVMF][ST][LIVMF][GSTAFY]
Pratt	90%	78%	0.83	GX(2)[FILMV]NP[AS]X[DST][FIL][AGP]
Teiresias	23%	89%	0.37	[ILMV]X(10)[ST]X(3)[ILMV]NX[AG]X(3)[AG]
Protomata	100%	87%	0.93	[ACGSTV]X[ACFGILMV]N[ACGPV][AGS][ACFG ILMV][DNST][ACFGILMV][ACGSTV]X[ACFGHI KLMTVWY]X(12)[FMY]X[ACFGHIKLMTVWY]X Q[ACFGHIKLMTVWY][ACFGILMV][AGS][AGS]

Le quorum (présenté à la section 4.3) est également un paramètre important laissé à la disposition de l'utilisateur. Le quorum est particulièrement corrélé à la taille des modèles obtenus en sortie et donc au nombre d'états présents dans l'automate.

La version Strictement SFP de cette section a fait l'objet d'un prototype avec implémentation de trois scores, ce qui nous a permis une première évaluation de leur efficacité (Coste et coll., 2004). Depuis les améliorations des performances de l'outil ont pu produire les résultats présentés ci-après et parus dans (Coste et Kerbellec, 2005).

Dans cette expérience, nous avons souhaité évaluer les performances du modèle le plus simple que notre approche puisse produire sur le jeu M des protéines MIP. Afin d'obtenir un modèle très simple et très général de la famille, nous règlons en premier le paramètre "Quorum" à son maximum qui est ici de 44 pour le jeu M. Ainsi, seules les zones supportant 100% des séquences de M deviennent caractéristiques et intégrées au modèle.
De plus, nous avons choisi de restreindre le motif à la première zone caractéristique obtenue par fusions des paires de fragments dans l'approche paire de fragments. Le faible niveau d'expressivité du modèle obtenu nous permet ainsi de le traduire en motif. Le motif, ou le pattern, est une signature courte et discrète de la famille (cf chapitre 1). La comparaison avec des méthodes de découvertes de motifs comme Pratt(Jonassen et Higgins, 1995) et Teiresias(Rigoutsos et Floratos, 1998) est alors possible. Nous utilisons les paramètres par défaut de ces deux méthodes, à l'exception du paramètre W (maximum length) de Teiresias qui a été fixé à 50 pour permettre la découverte de longs motifs. De plus nous utilisons la base de données Prosite qui nous fournit un motif de référence sur la famille des MIP.
Nous avons utilisé le programme Dialign2 avec les options suivantes : -nta -thr 5 -afc.

6.4.1 Résultats

Notons que le scan du motif MIP provenant de la base de motif Prosite retourne aussi bien des faux positifs que des faux négatifs. Le motif Prosite est ici à titre indicatif,

car il n'a pas été obtenu dans les mêmes conditions que les autres. Il a été produit de manière semi-automatique (programmes d'alignement et expertise humaine) à partir du jeu T.
Notre résultat, issu d'un apprentissage automatique depuis le jeu de séquence M, obtient pour sa part un rappel proche de celui du motif Prosite ainsi qu'une précision de 100%. De plus nous avons pu remarquer que les 16 séquences classées dans les faux-positifs de notre approche n'ont pas été validées MIP par expérience mais uniquement par similarité. Nous y trouvons également une demi-séquence, que notre approche considère à juste titre comme ne pouvant être fonctionnelle.
En comparaison avec les deux méthodes concurrentes que sont Pratt et Teiresias, nous constatons un gain très net de notre approche autant en rappel qu'en précision. Nous pouvons observer que tous les motifs ont fait émerger une zone connue comme étant le premier domaine "boîte NPA" chez les MIP. Notre motif relève un nombre de positions plus élevé que les autres. Lorsque l'on observe la structure on retrouve dans ce motif des positions conservées dues à leur importance au sein du canal formé par chaque protéine MIP. Ainsi les acides aminés appartenant à l'hélice alpha suivant la première boîte NPA et orientés vers l'intérieur du canal sont particulièrement bien conservés (Q par exemple).
Le comportement de notre approche dans une version simple montre de bonnes qualités de discrimination et de représentation sémantique sur un jeu connu comme possédant une similarité globale assez forte.
La prochaine section va nous permettre de nous focaliser sur une situation encore plus difficile pour une famille à forte similarité. La difficulté résidant dans la capacité à produire des signatures capables de discriminer une sous-famille par rapport à une autre lorsque la super-famille est à forte similarité.

6.5 Expérience de discrimination entre deux types de canaux MIP

Nous nous intéressons ici à l'apprentissage d'une sous-famille des MIP que nous avons nommé Water-Specific subfamily. La tâche consiste à apprendre des modèles capables de reconnaître ces canaux laissant passer strictement de l'eau, tout en maximisant la discrimination avec les autres canaux laissant passer du glycerol ou de petits solutés. Pour cela nous avons à dispositions un jeu de contre-exemples.
Cette expérience utilise également les options "-nta -thr 5 -afc" de Dialign2 pour produire les paires de fragments. Protomata est utilisé dans une version similaire à l'expérience précédente, mais cette fois-ci ce sont les 3 scores qui seront évalués. Enfin, nous avons utilisé l'expansion aux groupes physico-chimiques (voir les détails au chapitre 4) avec les réglages suivant : $\lambda_G = 10^{-7}$, $\lambda_\Sigma = 10^{-19}$ et les groupes correspondant aux différentes unions possibles des groupes de Taylor. Nous avons pu constater que sur une machine standard (3GHz) l'exécution de l'apprentissage n'a jamais excédé les 10 minutes pour un code qui n'était pourtant pas encore optimisé.
L'apprentissage se déroule autour d'un Leave-one-out à partir du jeu d'apprentissage

$W+$ et du jeu de contre-exemple $W-$, sachant que le jeu de contrôle C contre-exemple des MIP sera également parsé par chaque signature. Cette expérience nous permet d'étudier les qualités d'automates plus grands que de simple motifs. La contrainte étant que, pour rattraper une mauvaise précision qui sera engendrée par un parsing discret de nos signatures, nous utilisons une distance d'acceptation. La distance est produite par un parsing avec erreurs, et le seuil d'acceptation est donné par les contres-exemples.

6.5.1 Résultats

Nous présentons ici sur la figure 6.1 le résultat obtenu par l'utilisation de l'heuristique du score d'implication (score de Lerman prenant en compte les jeux négatifs) avec un quorum de 100%.

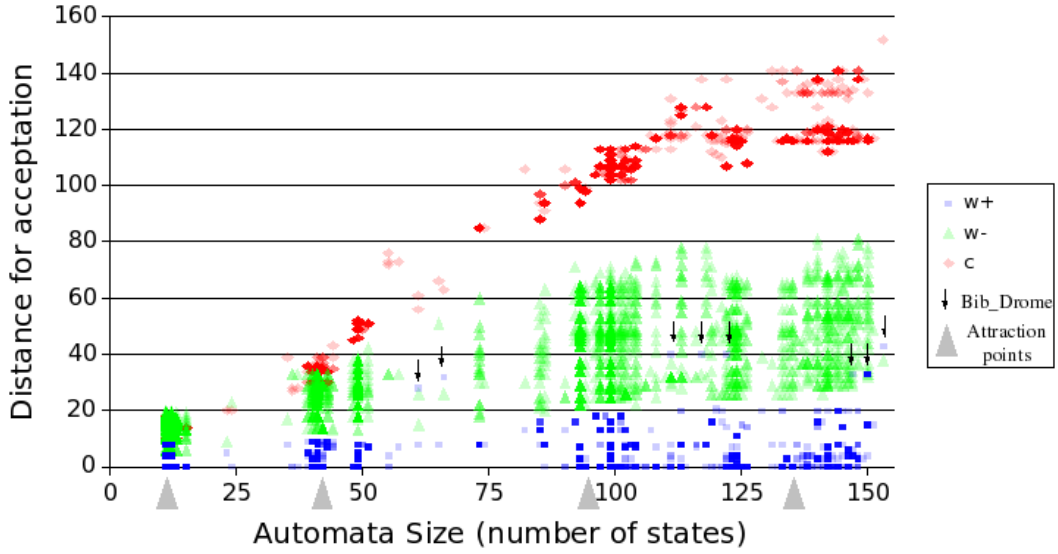


FIG. 6.1 – Distance de chaque séquence de validation pour son acceptation dans l'automate produit en utilisant l'heuristique d'implication. Leave-One-Out produit dans le but d'évaluer la caractérisation de $W+$ avec $W-$ comme contre exemple. Le jeu C est un jeu de contrôle (non MIP), seule sa séquence la plus proche est représentée.

Nous avons fait varier le nombre de paires de fragments utilisées pour produire les automates. Cette variation implique de fait une différence de la taille entre les automates (axe des abscisses). En effet, un grand nombre de fragments fusionnés entraînent l'apparition d'un nombre plus important de zones considérées comme caractéristiques. La figure montre une séparation manifeste lorsque l'on étudie la distance de chaque séquence vis-a-vis du modèle produit (axe des ordonnées). Sur l'axe représentant la taille des automates, nous avons indiqué 4 points d'attractions qui semblent liés à l'émergence progressive de zones caractéristiques. La première correspond au motif de la première

TAB. 6.2 – Performance de classification (W+ contre W-).

	Discret			Distance		
Taille des automates	Précision	Rappel	F-mes.	Précision	Rappel	F-mes.
10	100%	92%	0.96	100%	96%	0.98
40	100%	71%	0.83	100%	100%	1.00
100	100%	54%	0.70	100%	100%	1.00
130	100%	42%	0.59	100%	96%	0.98

boîte NPA mise en valeur dans l'expérience précédente. L'écart entre les jeux augmente progressivement avec la taille des automates, jusqu'à un point d'inflexion où la prise en compte d'un nombre plus important de fragments pour les fusions dans l'automate ne semble plus contribuer à une meilleure discrimination.
Seule la séquence *Bib_Drome* du jeu $W+$ est parfois notée à un niveau courant pour les séquences de $W-$. Cependant *Bib_Drome* est connue pour être divergente des autres MIP. Néanmoins, la distance nécessaire pour parser cette séquence est toujours plus faible que celle requise pour parser les séquence de $W-$ ou celles de C. Nous avons comparé dans les mêmes conditions les résultats produits par les autres heuristique de score sur des automates de taille environ 100 états (point d'inflexion). Les graphiques produits en section 6.2 montrent une meilleure discrimination de l'heuristique d'implication. Rappelons que cette heuristique est la seule à prendre en compte des séquences contre-exemple pour l'apprentissage Leave-One-Out. Nous voyons également un meilleur comportement de l'heuristique support dans ce type de situation. Nous retiendrons pour la suite que cette heuristique favorise le choix de fragments faisant partie de Blocs de PLMA.

Nous présentons également la table 6.2 qui rassemble les résultats de validation de l'apprentissage des automates produits par le score d'implication au niveau des différents points d'attractions avec deux types d'acceptations (discret ou distance).

L'approche a donc été capable de produire un résultat de 100% en précision et 100% en rappel pour des tailles d'automate comprises entre 40 et 100 états. L'intérêt de Promata a donc été mis en avant pour la discrimination entre sous-familles similaires. Le succès des heuristiques privilégiant les fragments liés à de nombreux autres fragments (support et implication) montre que de chercher à produire directement des ensembles de multiples fragments peut être envisagé, et c'est ce qui nous a amené à une formalisation du problème orientée Blocs de PLMA et aux expériences suivantes.

6.6 Expérience d'apprentissage de signatures de ligands TNF

Malgré les bons résultats obtenus sur les MIP, les premiers essais de la version Strictement SFP sur la famille des les ligands des TNF n'ont pas été concluants. En effet, pour une famille constituée de membres très peu similaires nous avons pu observer

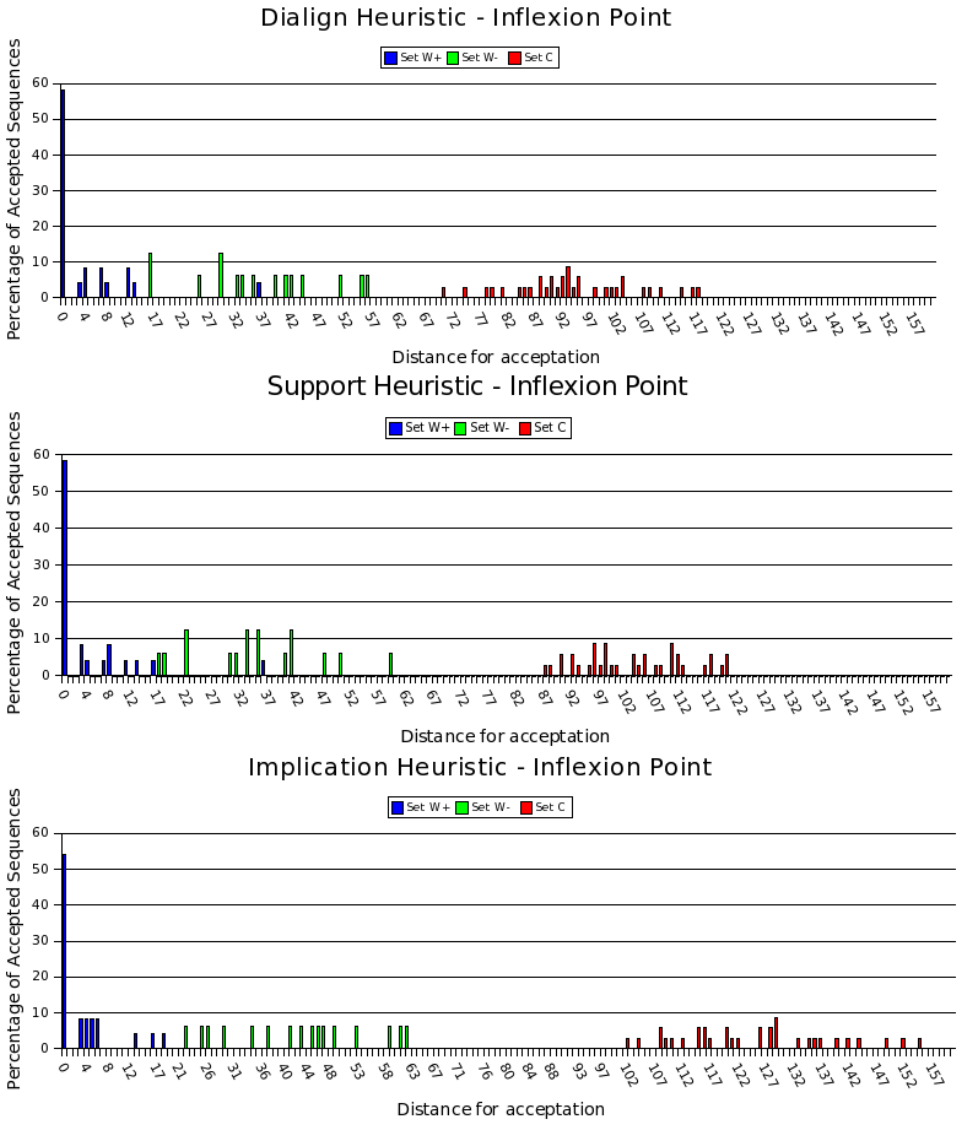


FIG. 6.2 – Distribution pour 3 heuristiques du pourcentage de séquences acceptées par un automate de taille approximativement 100 transitions.

que les modèles obtenus était sur-généralisés. Le jeu de la transitivité conduit parfois à la fusion de fragments totalement différents. C'est pourquoi, nous avons profité d'une refonte de l'implémentation pour essayer une expérience orientée vers l'identification de cliques de fragments. Nous avons ici implémenté la version Clique Semi-Exhaustive de l'approche.

De plus cette expérience nous a également permis d'observer les problèmes de court-circuit (au moment du parsing) des zones caractéristiques par des généralisations de

	Methode	Precision	Rappel	F-mesure
Parsing Discret				
	Prosite	0.75	0.67	0.71
	Teiresias	0	1	0
	Pratt	0.85	0.94	0.89
	Protomata-SFP Q=17	0.88	0.89	0.88
Parsing Distance				
	Pratt	0.86	1	0.92
Pratt Q=12	0.8	0.94	0.87	
Pratt Q=6	0.9	0.94	0.92	
Pratt Q=3	1	0.5	0.67	
Protomata-Cl Q=7	0.96	0.94	0.95	
	Protomata-Cl Q=6	**1**	**1**	**1**
	Protomata-Cl Q=5	1	0.94	0.97

TAB. 6.3 – Comparaison entre la version de Protomata Clique Semi-Exhaustif et d'autres méthodes sur un Leave-One-Out d'apprentissage de la famille des ligands des TNF. Q est la valeur du quorum.

zones de Gap. D'où l'idée de tester également le concept de chemins exceptions ne participant pas aux généralisations en Gap.
Nous avons donc effectué un Leave-One-Out sur le jeu positif de séquences $N+$ de la famille des TNF en prenant le jeu $N-$ comme jeu négatif.

Le tableau 6.3 rapporte les résultats obtenus par chaque méthode essayée. La première partie du tableau fait une comparaison sur des motifs obtenus par un quorum de 100% des séquences et un parsing direct. Notons que le premier motif n'est pas issu d'un apprentissage mais de la base de données Prosite. Pourtant issu d'une expertise semi-automatisée, le rappel associé n'est que de 67% sur ce jeu.
L'apprentissage à partir du programme Teireisias ne renvoie quant à lui qu'un motif pauvre et surgénéralisé. De son côté le programme Pratt produit un motif intéressant avec une F-mesure à 0.89. A titre de comparaison, nous avons également testé la version paire de fragments qui a produit un motif proche de celui de Pratt.
Pourtant, c'est bien en diminuant le quorum, ce qui rend l'utilisation du parsing distance indispensable, que nous avons pu montrer les meilleurs résultats en utilisant la version clique semi-exhaustive de Protomata. Afin de déterminer le seuil d'acceptation, nous avons pris la distance d'une séquence référence à laquelle nous avons appliqué un coefficient de 3/4 permettant une certaine marge.
Les paramètres de PRATT permettent d'obtenir des motifs pour des quorums différents. Cependant, la meilleure F-mesure obtenue n'est alors que de 0.92.
C'est donc la version Clique Semi-Exhaustive qui permet ici un apprentissage parfait de 100% de précision et 100% de rappel pour un quorum de 30%. Cette expérimentation permet d'illustrer l'importance d'avoir des chemins disjoints. Le graphique 6.3 nous permet de bien voir les évolutions croisées des courbes de précisions et de rappel pour cette version de Protomata.

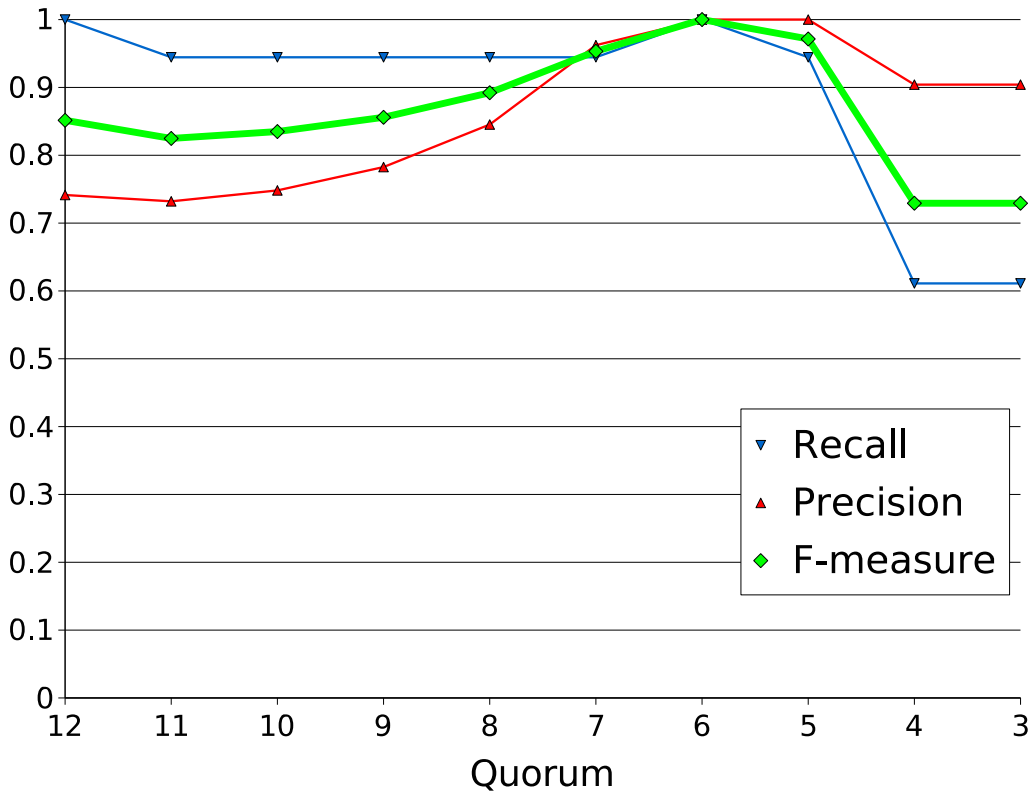


FIG. 6.3 – Impact du quorum sur la F-mesure.

6.7 Etude paramètrique des approches avec graines sur les ligands des TNF

6.7.1 Méthode

Les expériences présentées dans cette section ont pour but d'étudier la version Graines de Bloc de PLMA de notre approche. Nous avons pu vérifier sur une première implémentation que l'approche par fusion de fragments était efficace dans sa version paire de fragments sur une famille à forte similarité. Nous en avons dégagé que des Blocs de PLMA étaient construits de manière transitive. Dans une autre version, nous avons également constaté l'efficacité de la version Clique Semi-Exhaustive sur un jeu difficile.
A partir des deux premières implémentations, nous avons choisi une version de compromis. Cette version est résolument orientée vers la découverte de Bloc de PLMA. Elle se veut plus rapide qu'une approche exhaustive puisqu'elle se base sur une approche basée sur le tri de fragments qui servent de graine à la construction de Bloc de PLMA. Cette nouvelle implémentation nous permet alors de mettre en concurrence le choix d'un consensus fort et le choix d'un consensus faible (voir 3.8).

Le choix du jeu de séquences est également celui des ligands des TNF et le protocole expérimental reste identique à l'expérimentation précédente, c'est à dire un Leave-One-Out. Cependant, le choix du parsing a aussi évolué. Nous sommes désormais dans un parsing de poids et non plus en distance.
Nous avons effectué une étude plus systématique de l'influence de chaque paramètre choisi. Parmi les différents paramètres, nous avons celui du choix du consensus (faible ou fort). Nous avons également celui du seuil de Dialign fixé à 5 dans les versions précédentes. Enfin, nous avons évalué l'influence de l'intervalle de taille de fragments choisis.

6.7.2 Résultats

Les figures présentées dans cette section montrent l'évolution des courbes de précision, rappel et F-mesure en fonction du quorum pour des apprentissages d'automates avec la version "graine" de l'approche. Nous discutons la qualité des résultats et l'influence des paramètres.

La figure 6.4 montre un apprentissage effectué dans des conditions peu restrictives sur le nombre et le niveau de similarité entre les fragments. En effet, les bornes de taille 0 et 40 englobent l'ensembles des fragments fournis par Dialign. Le seuil fixé à 1 est également un seuil permettant de prendre en compte de nombreuses paires de fragments potentiellement très peu similaires. On constate que pour le choix d'un consensus fort l'approche parvient progressivement à améliorer le rappel en diminuant le quorum, et donc en augmentant la taille des automates produits. La F-mesure obtient son meilleur score pour un quorum de 9 séquences, puis diminue au fur et à mesure de la baisse de la précision.

Sur la figure 6.5 nous pouvons observer l'impact de l'augmentation du niveau du seuil de Dialign par rapport à la figure 6.4. L'approche bénéficie de moins d'information et les Blocs de PLMA de consensus fort ne se forment que pour des quorums bien plus petits, ce qui explique le décalage des courbes vers la droite.

Le fait de passer à un seuil dialign de 10 montre, sur la figure 6.6, un constat similaire à celui fait précédemment. Le peu de fragments pris en compte par un seuil de 10 permet des exécutions très rapides mais de mauvaise qualité, car basées sur trop peu d'information.
Prenons désormais pour la figure 6.7 les mêmes conditions que celles de la figure 6.4 en ne changeant que le type de consensus en passant d'un consensus fort à un consensus faible. Nous constatons alors l'importance du choix d'un consensus fort dans le cadre de ce jeu très hétérogène des TNF. Compte tenu de paramètres peu sélectifs au niveau du choix des fragments, la transitivité contenue dans les Blocs de PLMA ne permet pas de préserver les informations importantes.

Par rapport à la figure 6.7, les figures 6.8 et 6.9 ne montrent pas d'améliorations très importantes pour des variations du seuil Dialign.

La figure 6.10 présente des apprentissages produits dans le cadre d'une restriction faite sur la taille des fragments pris en compte. Le choix d'un consensus fort et l'augmentation progressive du seuil Dialign sur les figures 6.11 et 6.12 ne permettent pas l'obtention de bons résultats pour ce choix de bornes.

C'est sur la figure 6.13 que nous pouvons observer les meilleurs résultats avec une F-mesure à 100% pour des quorums entre 6 et 9. Nous voyons ici l'importance du choix des fragments pris en compte dans le cadre de la recherche de Blocs de PLMA de consensus faible. La figure 6.8 avait montré un comportement qui pouvait tendre à une amélioration vers le quorum 8, mais l'expérience de la figure 6.13 nous indique que c'étaient surtout les fragments de grande taille qui venaient perturber l'émergence de zones caractéristiques intéressantes.

Compte tenu du faible nombre de paires de fragments de taille inférieure à 15 et de significativité supérieure à 5, les figures 6.14 et 6.15 montrent des résultats qui se dégradent.
Nous voyons à présent sur les figures 6.16, 6.17 et 6.18 qu'il n'y a pas beaucoup d'intérêt à utiliser le choix du consensus fort pour de très petits fragments.
Même avec le choix d'un consensus faible, les graphiques 6.19, 6.20 et 6.21 ne semblent pas meilleurs que les précédents pour des tailles de fragments très petites.

Si le choix de la taille des fragments considérés s'effectue pour des tailles moyennes entre 10 et 30 acides aminés, le graphique 6.22 montre un comportement correct du consensus fort pour un seuil dialign de 1. Cependant les résultats s'effondrent toujours aussi rapidement avec l'augmentation du seuil Dialign sur les graphiques 6.23 et 6.24
Nous finissons par la présentation du comportement de l'approche dans le cadre de choix de Bloc de PLMA à consensus faible sur une taille moyenne de fragments. Les 3 résultats obtenus sur les figures 6.25, 6.26 et 6.27 montrent tous un pic de F-mesure très bonne qualité qui se déplace sensiblement sur l'axe du quorum en fonction du seuil Dialign choisi.

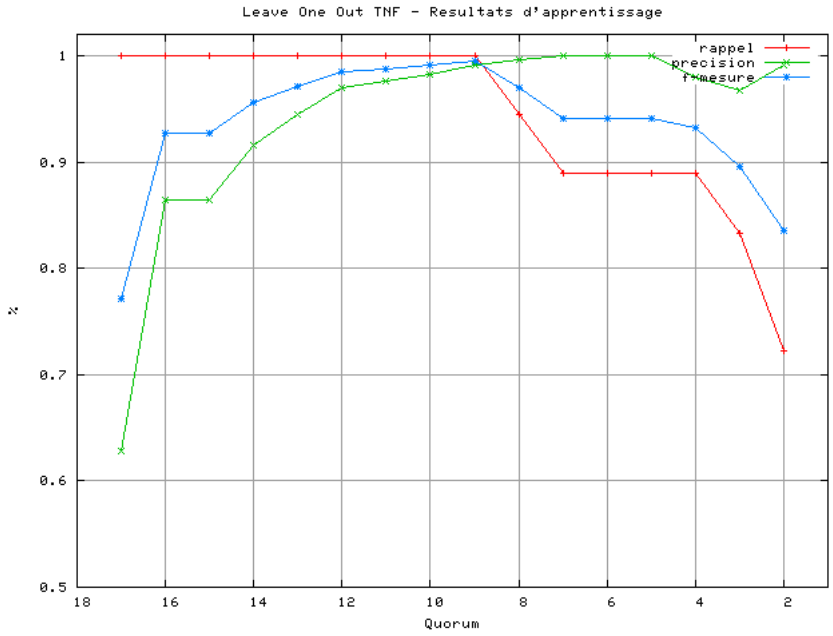


FIG. 6.4 – Consensus Fort, Taille des fragments entre 0 et 40, Seuil Dialign à 1

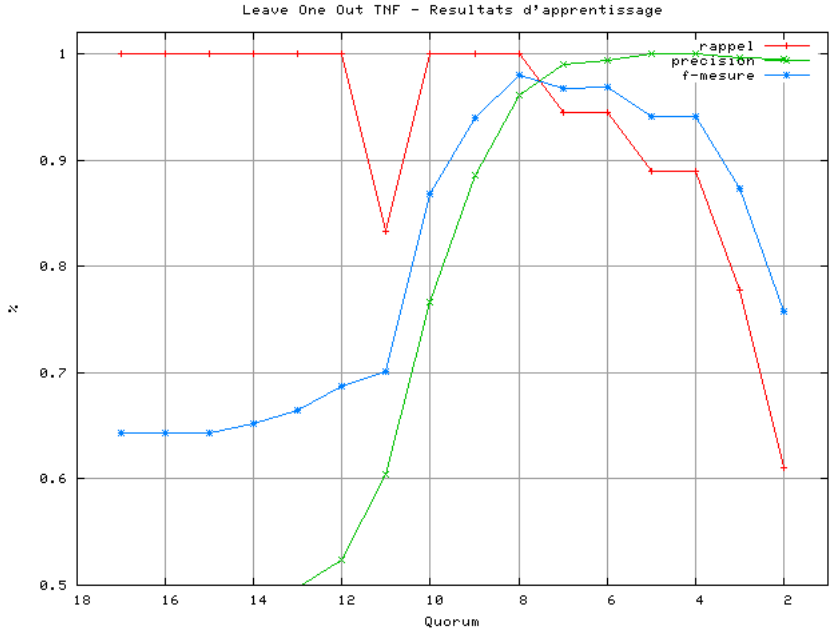


FIG. 6.5 – Consensus Fort, Taille des fragments entre 0 et 40, Seuil Dialign à 5

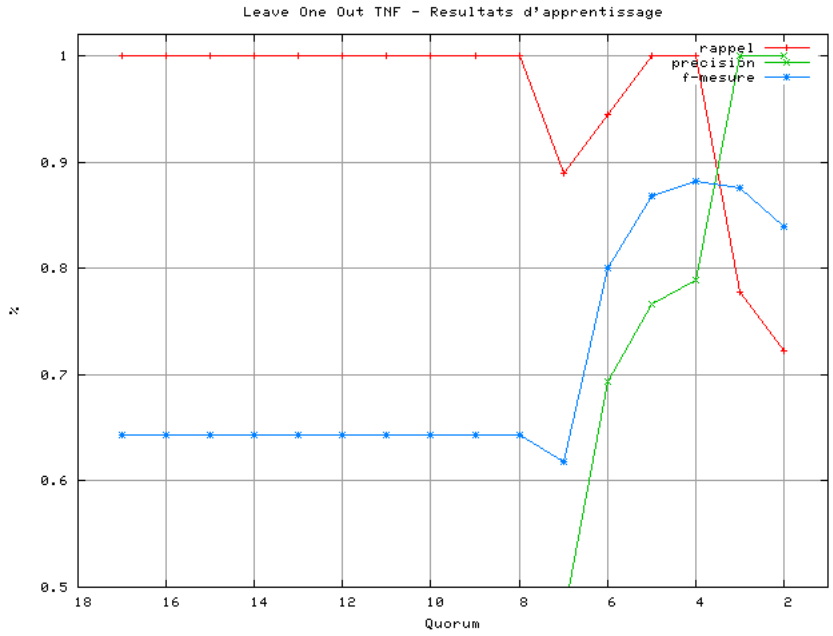


FIG. 6.6 – Consensus Fort, Taille des fragments entre 0 et 40, Seuil Dialign à 10

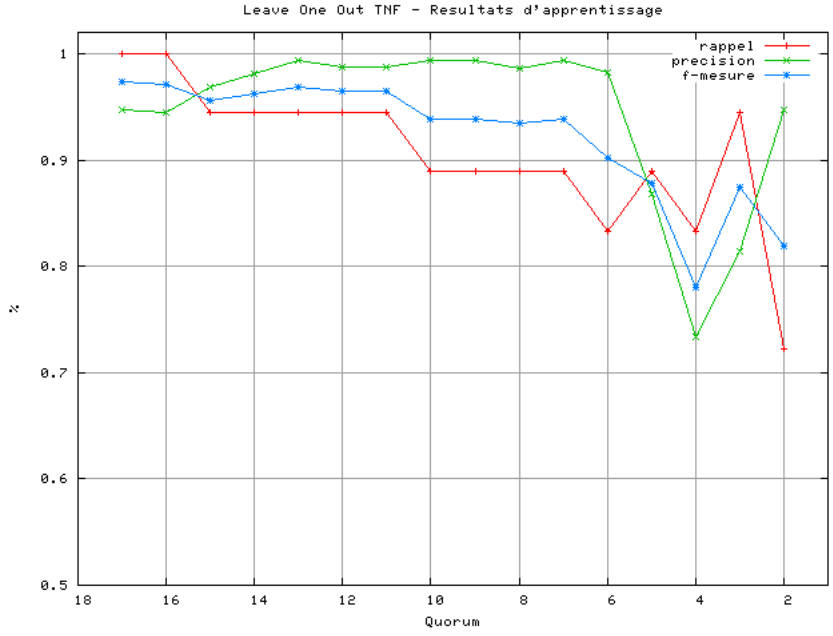


FIG. 6.7 – Consensus Faible, Taille des fragments entre 0 et 40, Seuil Dialign à 1

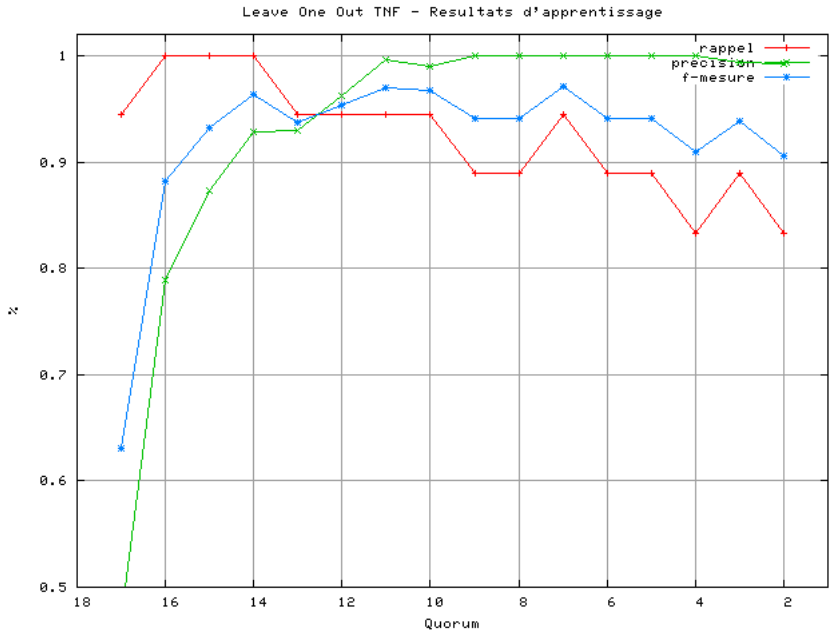


FIG. 6.8 – Consensus Faible, Taille des fragments entre 0 et 40, Seuil Dialign à 5

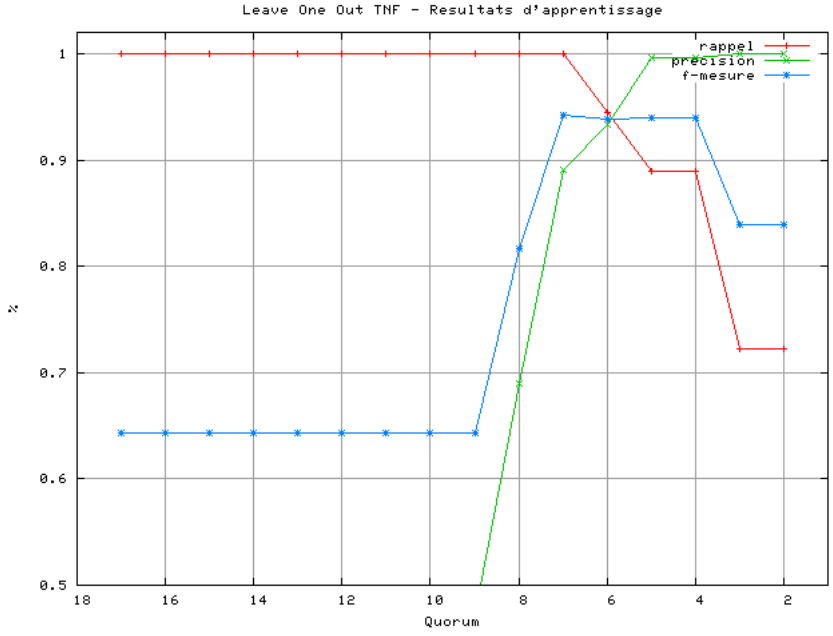


FIG. 6.9 – Consensus Faible, Taille des fragments entre 0 et 40, Seuil Dialign à 10

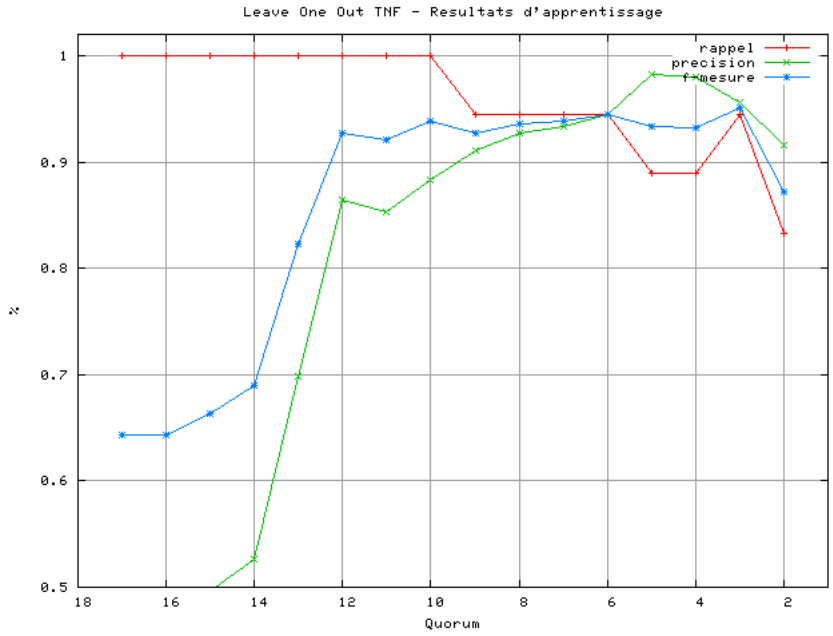


FIG. 6.10 – Consensus Fort, Taille des fragments entre 0 et 15, Seuil Dialign à 1

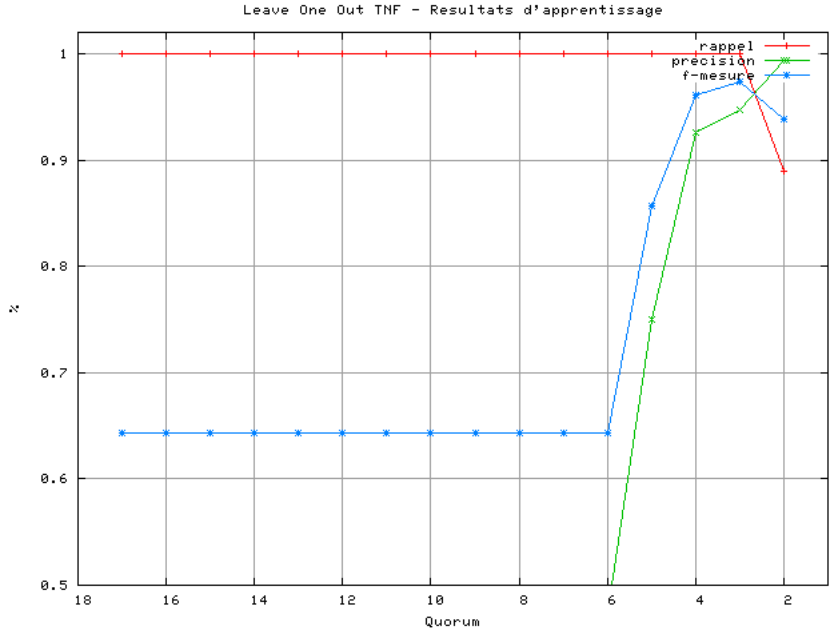


FIG. 6.11 – Consensus Fort, Taille des fragments entre 0 et 15, Seuil Dialign à 5

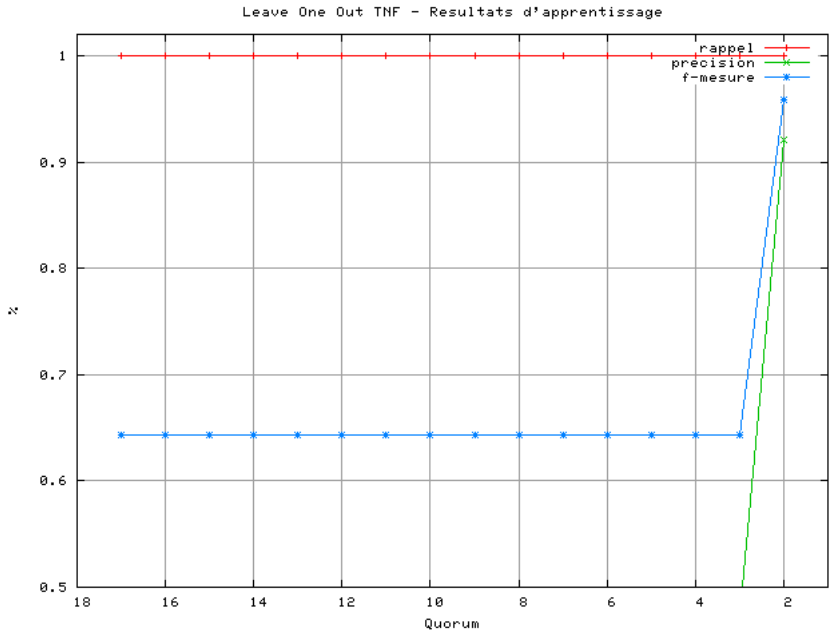


FIG. 6.12 – Consensus Fort, Taille des fragments entre 0 et 15, Seuil Dialign à 10

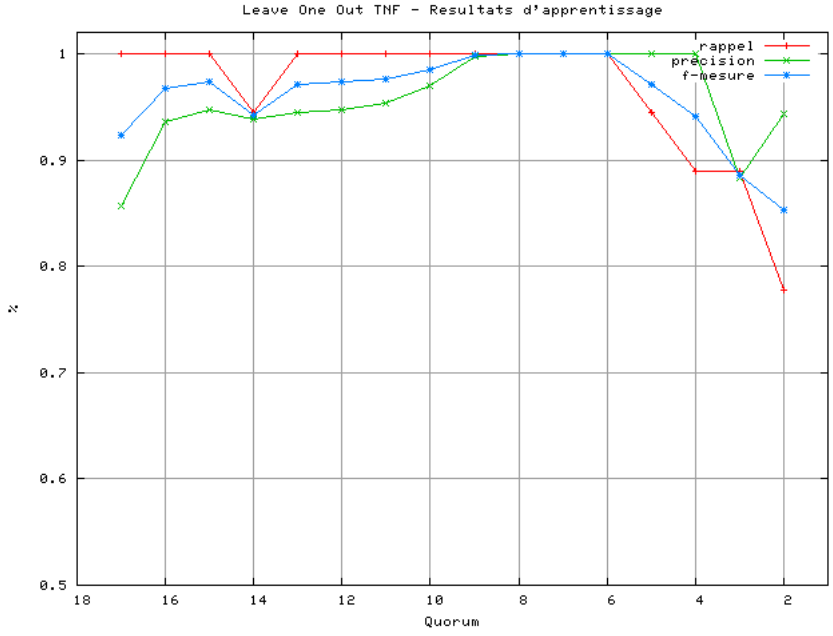


FIG. 6.13 – Consensus Faible, Taille des fragments entre 0 et 15, Seuil Dialign à 1

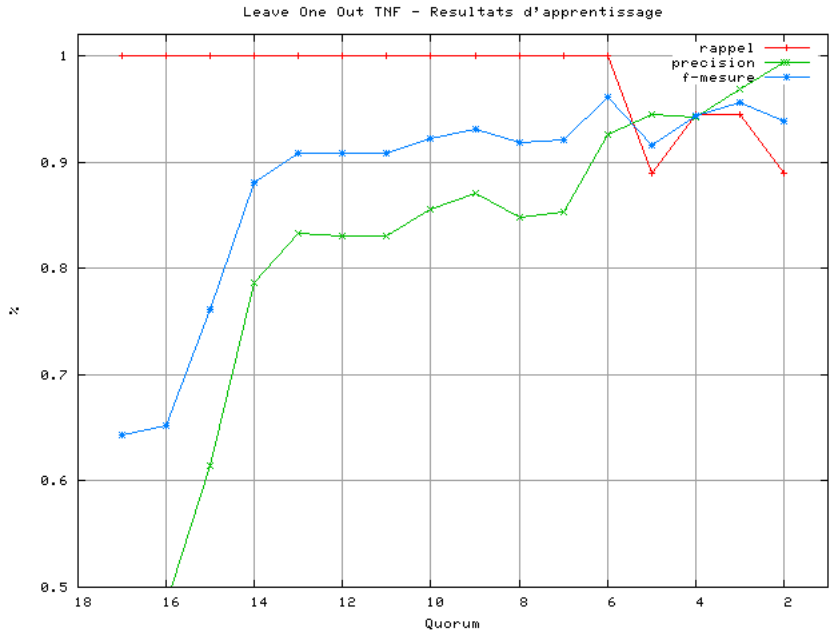


FIG. 6.14 – Consensus Faible, Taille des fragments entre 0 et 15, Seuil Dialign à 5

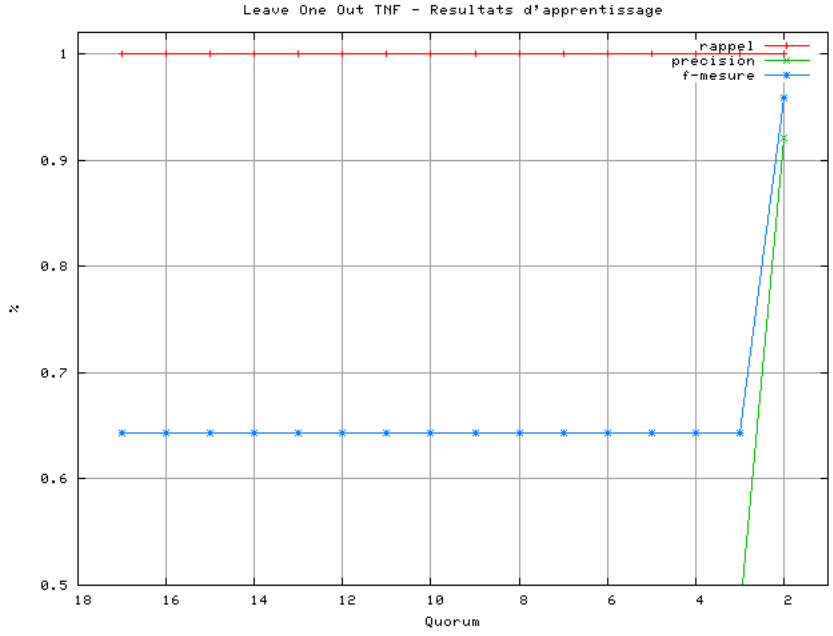


FIG. 6.15 – Consensus Faible, Taille des fragments entre 0 et 15, Seuil Dialign à 10

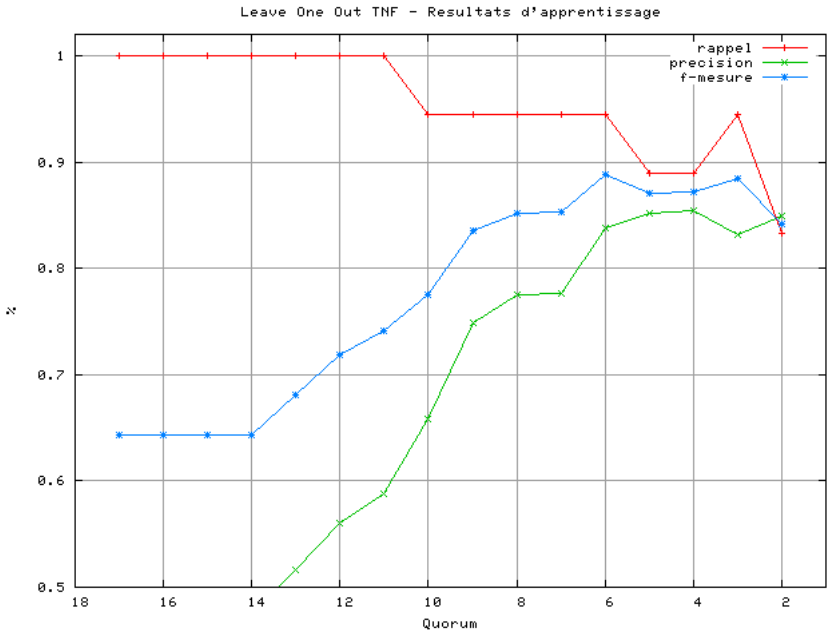


FIG. 6.16 – Consensus Fort, Taille des fragments entre 0 et 10, Seuil Dialign à 1

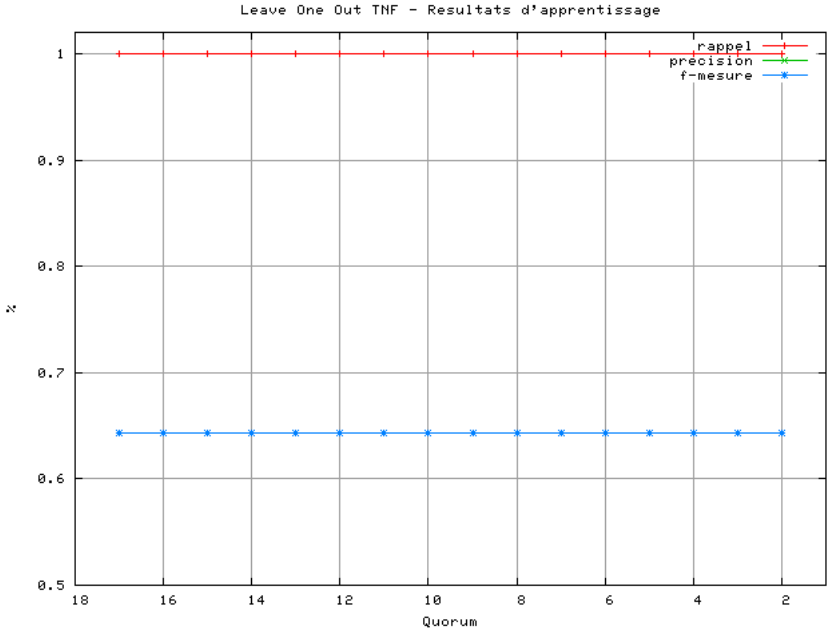


FIG. 6.17 – Consensus Fort, Taille des fragments entre 0 et 10, Seuil Dialign à 5

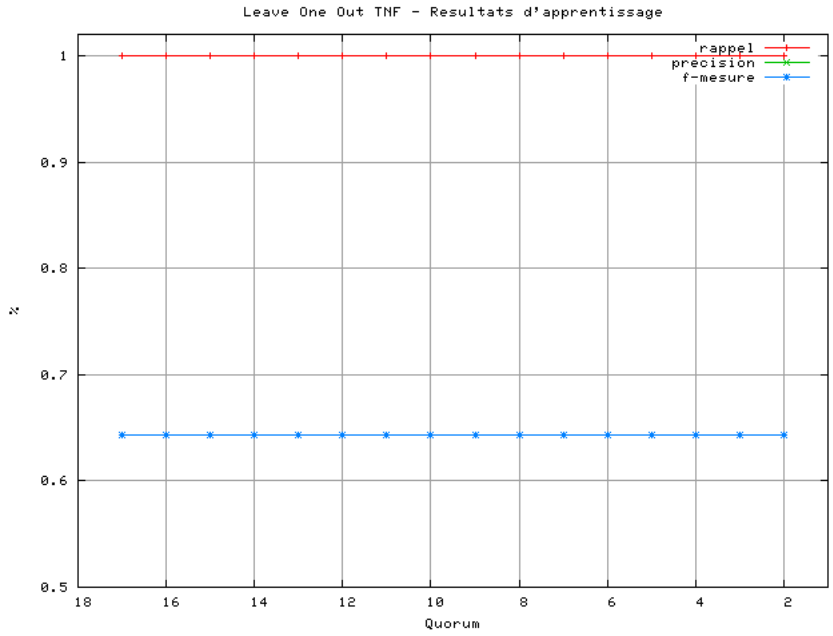


FIG. 6.18 – Consensus Fort, Taille des fragments entre 0 et 10, Seuil Dialign à 10

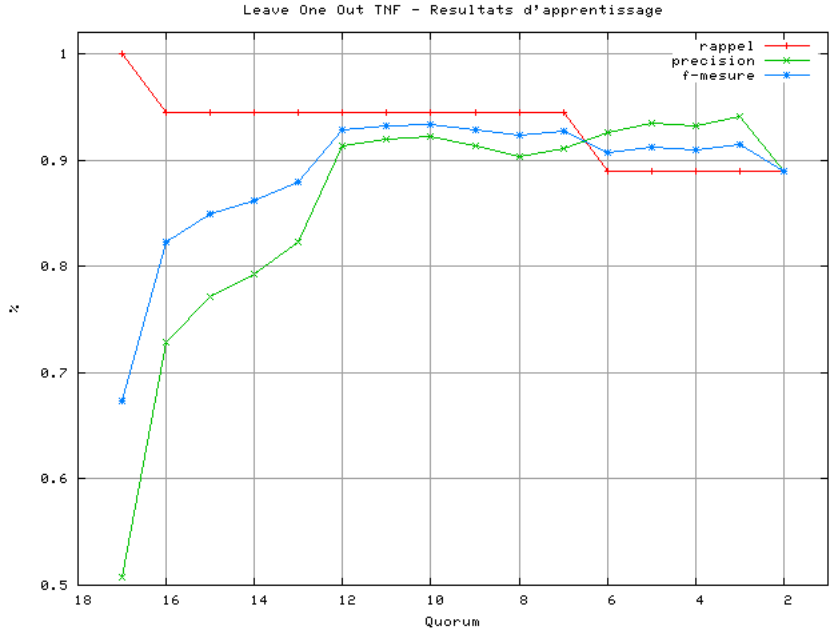


FIG. 6.19 – Consensus Faible, Taille des fragments entre 0 et 10, Seuil Dialign à 1

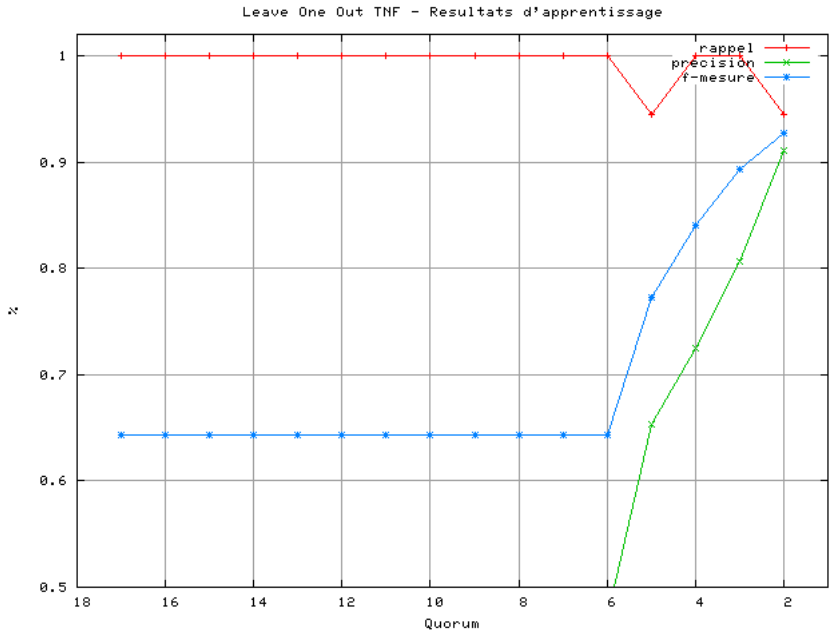


FIG. 6.20 – Consensus Faible, Taille des fragments entre 0 et 10, Seuil Dialign à 5

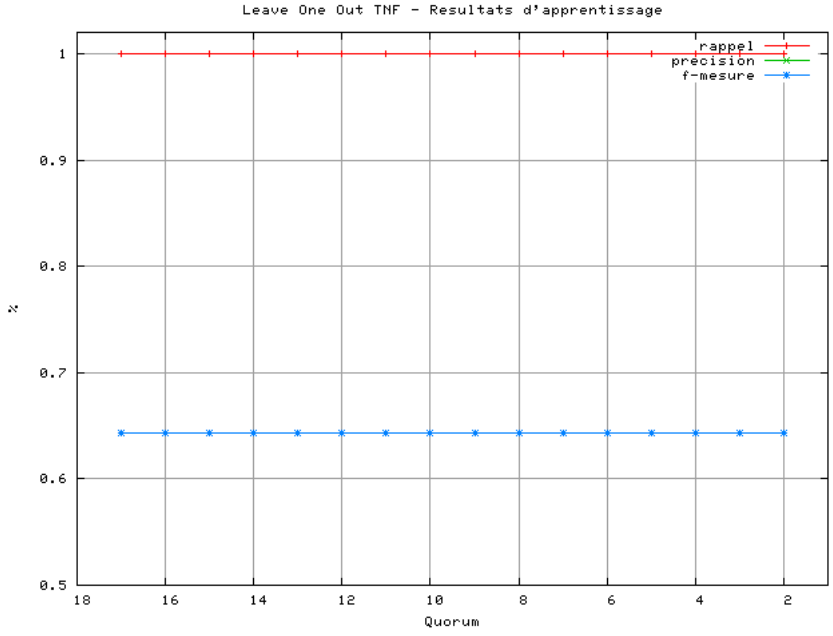


FIG. 6.21 – Consensus Faible, Taille des fragments entre 0 et 10, Seuil Dialign à 10

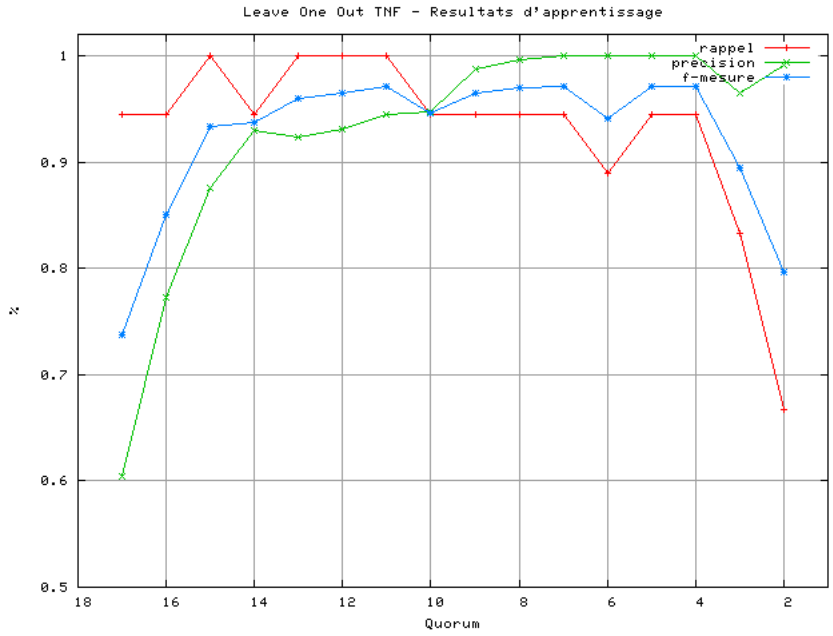


FIG. 6.22 – Consensus Fort, Taille des fragments entre 10 et 30, Seuil Dialign à 1

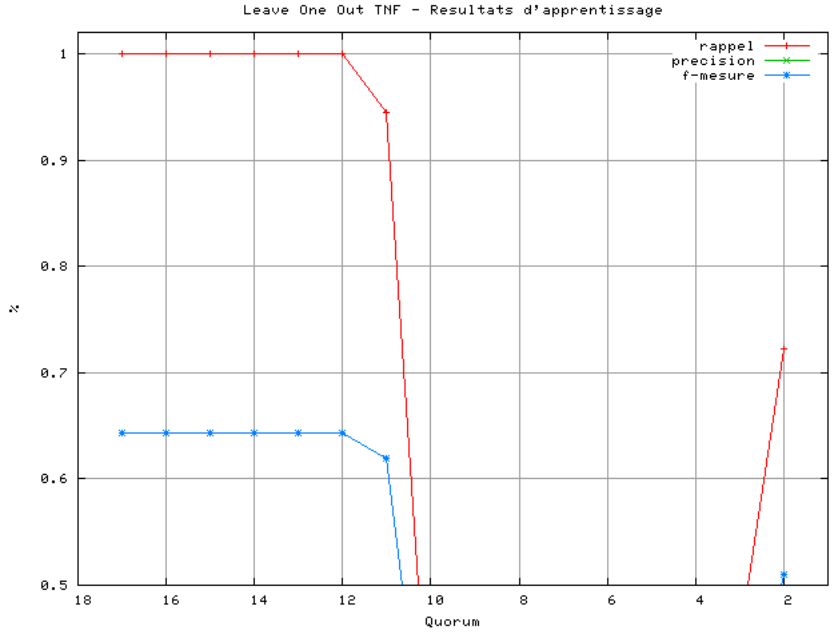


FIG. 6.23 – Consensus Fort, Taille des fragments entre 10 et 30, Seuil Dialign à 5

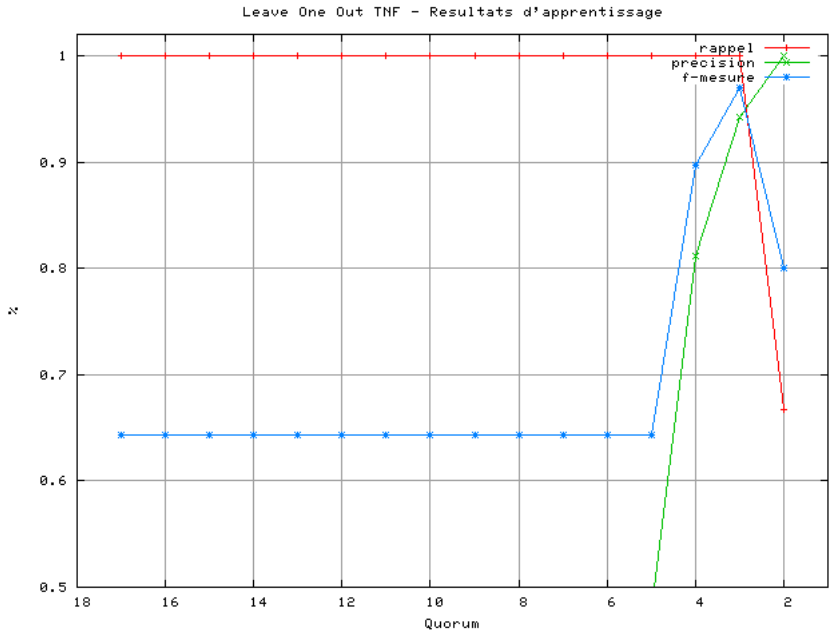


FIG. 6.24 – Consensus Fort, Taille des fragments entre 10 et 30, Seuil Dialign à 10

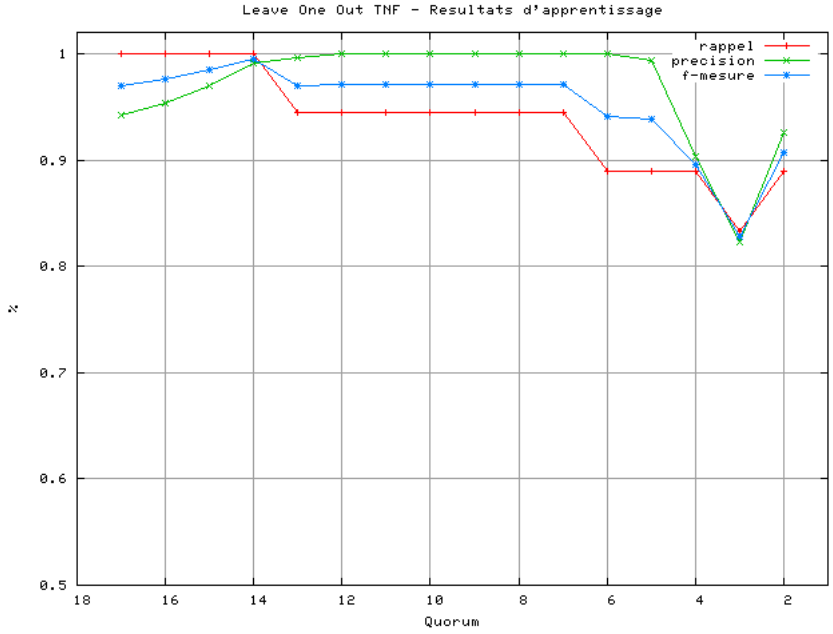


FIG. 6.25 – Consensus Faible, Taille des fragments entre 10 et 30, Seuil Dialign à 1

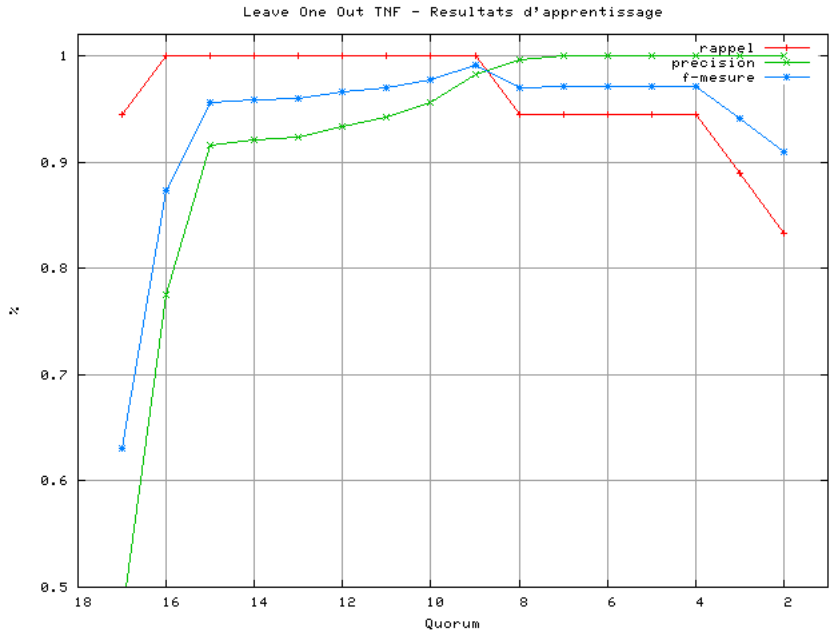


FIG. 6.26 – Consensus Faible, Taille des fragments entre 10 et 30, Seuil Dialign à 5

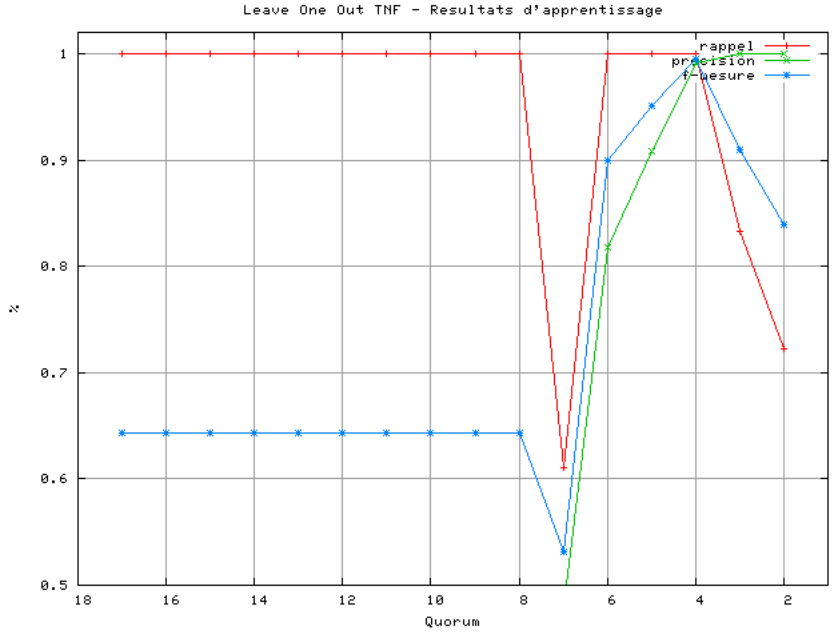


FIG. 6.27 – Consensus Faible, Taille des fragments entre 10 et 30, Seuil Dialign à 10

6.7.3 Conclusion

Cette étude aura permis d'observer l'intérêt du choix de consensus forts lorsque les fragments ne sont pas choisis avec précision. Elle aura également montré que l'approche appliquée sur des fragments de taille assez faible permet aux Blocs de consensus faible d'obtenir de très bons résultats. Enfin, les variations de quorum permettent de confirmer les hypothèses introduites dans notre approche. Les meilleurs résultats ne sont obtenus ni pour de petits modèles de type motifs, ni pour des modèles trop large et surspécialisés.

6.8 Courbe MDL

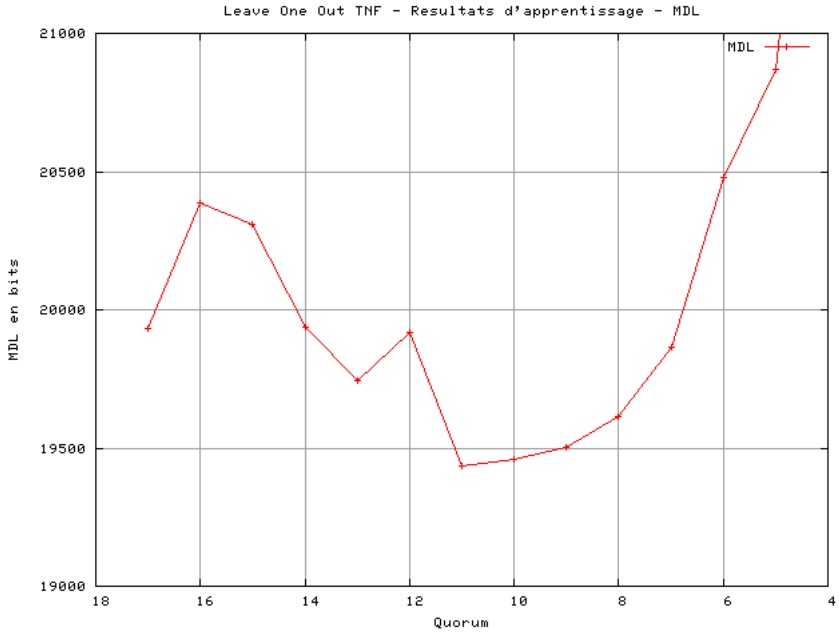


FIG. 6.28 – Courbe MDL

La courbe 6.28 a été produite lors de l'expérience d'apprentissage de modèles sur le jeu des TNF avec les paramètres ayant produit la figure 6.4. Nous observons que la zone de meilleur compression se situe entre un quorum de 8 et un quorum de 11. Elle correspond également à une zone de bons résultats d'apprentissage.
Cette mesure n'est pas toujours corrélée à de bons résultats d'apprentissage, ni même à l'idée de ce que l'utilisateur présente comme le meilleur compromis de modèle représentant sa famille d'intérêt. Cependant, c'est généralement une bonne indication permettant d'apprécier la qualité du modèle obtenu sous l'angle de la théorie de l'information.

6.9 Scan de Swissprot

Nous avons expérimenté le scan de la banque Swissprot à partir d'un modèle produit avec le jeu de ligands des TNF en choisissant l'approche avec graines et les paramètres suivants : consensus fort, taille des fragments comprise entre 0 et 40, seuil dialign à 1, et quorum à 8. Ce choix est inspiré du graphique 6.4.
Le résultat, selon le deuxième type de score (non disponible pour le moment sur l'interface web) est le suivant :

- Entre un score de 102 à 43 il n'existe que des séquences de TNF
- Le nombre de séquences de ligand de TNF de score supérieur à 43 est de 98
- Le nombre de séquences de ligand de TNF inférieur à 43 et supérieur à 0 est de 9
- Sur les 9, il y en a 4 dont le score est supérieur à 33
- Les dernières sont attribuées au membre numéro 4 des ligands des TNF

6.10 Conclusion

Ce chapitre relatif aux expérimentations montre l'évolution historique de l'approche. La première avancée importante a été l'introduction de contrainte de préservations. En effet, à partir de cette étape nous avons été capables d'obtenir des résultats intéressants sur une famille à forte similarité comme la famille des MIP. Même si elle n'a pas toujours été reprise par la suite, la mesure ayant permis d'effectuer des discriminations entre sous-familles appartenant à une super-famille à forte similarité a aussi rempli un rôle très important.
Ensuite, ce sont les idées de quorum, puis d'exception qui ont permis d'obtenir des représentations bien adaptées à des jeux plus difficiles, car à faible similarité, tel que la famille des ligands des TNF.
Cependant les représentations sous forme d'automate, appréciées par leur lisibilité, souffrent du manque de séquences et donc d'information au niveau des variations possibles en acides aminés pour chaque positions (avec 5 séquences il est impossible d'explorer les 13 acides aminés hydrophobes possibles à certaines positions). Nous avons donc à plusieurs reprises compensé ces problèmes par l'utilisation d'un outil d'extension à des groupes physico-chimiques ou bien par des méthodes de score de parsing basées sur la fréquence d'apparition de chaque acide aminé.
Même s'il faut revenir à des mesures probabilistes de type HMM, nous avons montré que nous étions capables de produire un *squelette* bien plus expressif et proche de la réalité de la structure de la famille que ne l'est un profil HMM.
Le résultat de scan de l'automate des ligands de TNF sur prosite est intéressant et nous incite à reitérer l'opération sur des banques d'EST afin de retrouver des séquences non-annotées et potentiellement membres de cette famille.

Conclusion

6.11 Avancées obtenues par notre approche

En choisissant de produire une nouvelle approche d'apprentissage de signatures de familles de protéines, nous avons dû répondre à de nombreux problèmes.
Tout d'abord, nous avons replacé toute la problématique habituelle de découverte de motifs dans le cadre plus large de la théorie des langages. A partir de là, nous avons pu observer les limites en expressivité des techniques existantes.
C'est ainsi que nous avons décidé de nous tourner vers les grammaires rationnelles et leurs représentations sous forme d'automates. Nous étions conscients que les algorithmes courants d'inférence grammaticale par fusion d'états montraient, d'un côté de bons résultats théoriques avec des automates finis déterministes (DFA), mais, d'un autre côté ni les DFA, ni les NFA (automates finis non-déterministes, pourtant mieux adaptés à notre application) n'avaient encore véritablement montré de bonnes capacités d'apprentissage sur des échantillons de séquences réelles de manière générale et, encore moins, sur des échantillons de séquences appartenant à une famille protéique. C'est pourtant ce que notre approche a montré sur différents jeux de protéines particulièrement hétérogènes au niveau de leurs fonctions, similarités et structures tridimensionnelles.
Un problème majeur était de trouver une méthode d'alignement de séquences se prêtant beaucoup mieux à l'émergence de blocs qui regroupent des fragments de séquences appartenant à un sous-ensemble du jeu d'apprentissage, ce que permettait le modèle de type NFA, capable de représenter des compositions de zones similaires et des disjonctions en sous-familles distinctes. C'est-à-dire que nous avons souhaité sortir d'un alignement multiple global traditionnel, ainsi que des alignements de domaines hyperconservés, pour aller vers des alignements locaux partiels et multiples (PLMA). Malgré différents problèmes liés à la complexité de la gestion des fragments dans une telle approche, le fait de projeter dans un graphe notre problème de recherche de Blocs de PLMA nous a permis d'obtenir les résultats escomptés.
L'identification des problèmes de bruit engendrés par la superposition de Blocs de faible similarité a été le premier élément qui, une fois résolu par l'introduction de différentes contraintes, a permis, à la phase de généralisation des NFA, de produire des modèles performants.
Les NFA se sont en effet montrés aptes et robustes, dans le cadre de notre approche, à représenter des familles de séquences de protéines. Nous avons donc appliqué des méthodes issues d'algorithmes par fusion d'états, en se basant cette fois-ci sur la fusion

de fragments qui permet une meilleure identification des contextes et de la sémantique associée à différentes positions.
L'application biologique nous a parfois entraînés sur des concepts spécifiques comme la représentation des zones non-similaires par des états uniques représentant des gaps, ou bien la mise en valeur de séquences considérées comme étant des exceptions à des zones caractéristiques, ou encore un outil permettant d'étendre des ensembles d'acides aminés à des groupes physico-chimiques connus.
Les expériences et les moyens ont été mis en œuvre pour les produire et rendre accessible un programme qui a permis de rendre compte de la qualité des modèles obtenus. Les soins apportés à la génération des jeux d'apprentissage et au processus de validation laissent entendre que notre approche produit des signatures robustes, capables de représenter des familles de protéines de façon discrète, et capables de faire valoir des performances en reconnaissance de nouveaux membres et en discrimination avec d'autres familles.
Enfin, nous avons adapté deux types de mesures particulièrement utiles dans notre approche. La première mesure est un score permettant un parsing fin de séquences dans les automates. La seconde est un score de type MDL (minimum description length) permettant de donner une estimation de la capacité de compression que possède un automate par rapport aux séquences de l'échantillon. Le choix d'automates non déterministes (NFA) permet une bonne représentation sémantique de domaines et d'enchaînements de ces domaines.

6.12 Perspectives

La version Protomata-Learner se présente actuellement comme le meilleur compromis de notre approche et devient un nouvel outil de bioinformatique à part entière. Nous envisageons à terme de procéder à des apprentissages massifs de familles de protéines afin de construire une banque comparable à Prosite, Pfam, ou encore Prints, et pourquoi ne pas envisager de l'intégrer à Interpro ? Nous pourrions être particulièrement novateurs en montrant des modèles de super-familles incluant les différentes compositions de sous-familles, à la manière des Clans de Pfam, mais de manière observable sur des automates.
Cependant, il faudra augmenter la performance des algorithmes, comme en faisant éventuellement appel à des techniques de recherche opérationnelle sur le problème de la recherche d'alignements de type PLMA. De même, les nouveaux concepts introduits, comme les exceptions par exemple, sont des sujets à perfectionner, approfondir et pourquoi pas généraliser à d'autres approches.
La dernière méthode d'évaluation du score d'un parsing des séquences dans les automates nous a rapprochés des évaluations que proposaient les HMM profils. En effet, notre score tient compte des fréquences d'apparition des acides aminés. Cependant, notre approche permet l'apprentissage d'automates sur une famille de protéines et permet par conséquent l'inférence du squelette d'un véritable HMM et non plus d'un simple

HMM profil. Afin d'obtenir ce que l'on pourra véritablement qualifier de HMM, il faudra encore quelques études permettant une calibration plus poussée de notre score.
Ce que nous avons pu produire avec un apprentissage par fusion d'états montre qu'il est possible d'obtenir des résultats sur des données réelles avec ce type d'approche. Nous pensons à y évaluer différentes variantes, comme l'utilisation d'automates typés étudiés précédemment par Daniel Fredouille.
Nous avions, entre autres applications directes, une collaboration avec l'UPRES Hématologie et Immunologie sur la famille des TNFs (Tumor Necrosis Factor) impliqués dans le développement de cancers. Les résultats obtenus, tant en prédiction qu'en modélisation, sont très stimulants. Nous attendons tout particulièrement les premiers scans sur des banques EST pour étudier de nouveaux candidats.
Nous avons ainsi soulevé un intérêt particulier vis-à-vis des résultats obtenus, tant en terme de reconnaissance de nouvelles séquences qu'en tant que modélisation de propriétés fonctionnelles ou structurales. Nous discuterons alors les différents protocoles qui pourraient permettre des validations en *wet-lab* et les retours possibles que cela impliquerait au niveau de l'approche.
En parallèle, il sera aussi nécessaire d'améliorer la visualisation des séquences acceptées par le modèle dont on aura, par ailleurs, eu l'occasion d'étiqueter certaines zones par diverses annotations tirées de la littérature.

6.13 Faut-il aller vers plus d'expressivité ?

Au cours des différentes expériences et des connaissances acquises sur les familles de protéines, leurs séquences, leurs structures et les modèles les représentant, nous avons remarqué que les langages réguliers apparaissent adaptés à la description de familles de protéines. Les automates ont un niveau d'expressivité suffisant pour représenter des sites actifs et la façon dont ils s'enchaînent. Cependant, il existe certaines limites. Ces langages ne permettent pas de mettre en relief des phénomènes de domaines répétés par exemple.
Il paraît donc intéressant d'imaginer des modèles encore plus expressifs pour les séquences protéiques. Cependant, nous avons surtout constaté que c'est plus particulièrement au niveau des séquences nucléiques d'ADN ou d'ARN que l'on peut observer des phénomènes nécessitant véritablement de passer à un niveau de langage supérieur. Notons par exemple la tige-boucle de l'ARN qui nécessite une représentation de type palindrome ; ils relèvent donc des langages algébriques. Une structure existant dans l'ARN, le pseudo-noeud, met en jeu des structures encore plus complexes correspondant à la fois à des phénomènes syntaxiques de répétitions et de palindromes, ils relèvent ainsi des langages contextuels. La figure 6.13 tirée de (Searls, 1997), représente les différentes classes de langages nécessaires à la prise en compte de tels phénomènes. Il existe également des travaux appliqués à l'inférence sur séquences d'ARN concernant des représentations hors-contexte (Sakakibara et coll., 1994).

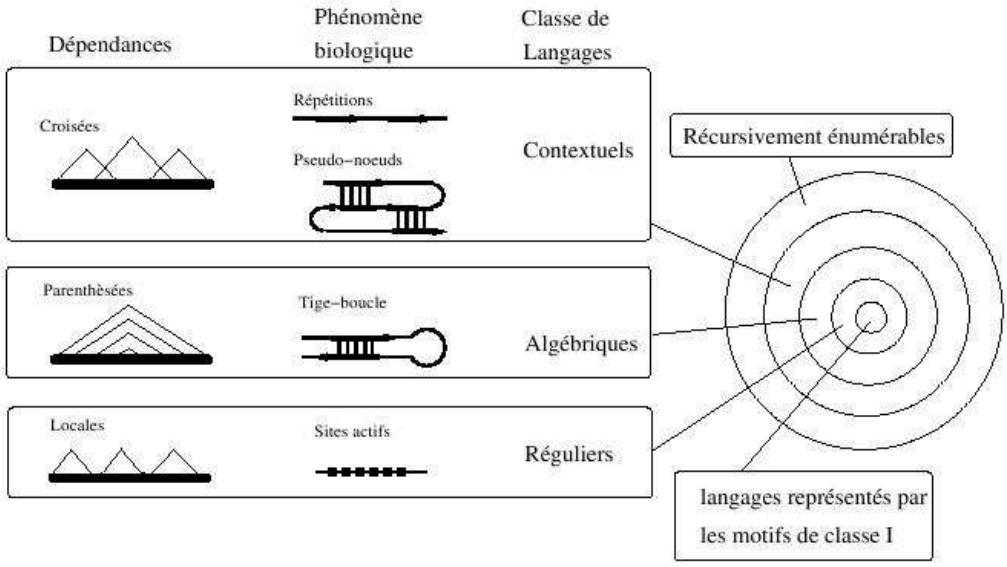


FIG. 6.29 – Classes de langages et séquences biologiques (Searls, 1997).

Malgré tout, cette thèse a été l'occasion de montrer que les familles répertoriées dans les banques sont souvent une vision biaisée par le fait qu'il existe des motifs ou des profils capables de les représenter. En effet, que ce soit par l'apparition récente des clans Pfam ou par de nouvelles définitions de super-familles, nous constatons que l'augmentation du pouvoir d'expressivité des modèles permet de poser des problèmes évités jusqu'ici, car considérés comme flous.
Il est donc possible d'espérer voir émerger des propriétés particulièrement intéressantes sur des modèles qui seraient le fruit d'un apprentissage de grammaires toujours plus expressives.

Annexe A

Annexe

Le web est l'ami du lecteur

Au cours de cette thèse nous avons évoqué à plusieurs reprises des séquences appartenant à des familles de protéines. Nous invitons le lecteur à se familiariser à ces séquences et leurs annotations au travers des différentes banques de données publiques disponibles sur internet.
De même il pourra à loisir utiliser les jeux que nous mettons à disposition afin de tester les outils d'alignement multiple, de découverte de motif et bien sûr le serveur de Protomata-Learner (implémentation de notre approche).
Voici quelques liens qui permettront une certaine interactivité et une meilleure prise en main du manuscrit :
http ://www.expasy.uniprot.org (base de donnée de séquences)
http ://www.rcsb.org/pdb (base de donnée de structure)
http ://www.ebi.ac.uk/Tools/clustalw2 (programme d'alignement)
http ://www.ebi.ac.uk/pratt (programme de découverte de motif)
http ://protomata-learner.genouest.org (interface de notre approche)
http ://www.kerbellec.com/These_bioinformatique (source de données sur la thèse)

Bibliographie

Abdeddaïm, S. et Morgenstern, B. (2000). Speeding up the dialign multiple alignment program by using the 'greedy alignment of biological sequences library' (gabios-lib). Dans *JOBIM*, pages 1–11.

Agre, P. et Kozono, D. (2003). Aquaporin water channels : molecular mechanisms for human diseases. *FEBS Lett.*, pages 72–78.

Altschul, S., Gish, W., Miller, E. et Lipman, D. (1990). A basic local alignment search tool. *Journal of Molecular Biology*, 215:403–410.

Angluin, D. (1987). Learning regular sets from queries and counterexamples. Dans *Information and computation*, pages 87–106.

Ashkenazi, A. (2002). Targeting death and decoy receptors of the tumour-necrosis factor superfamily. *Nature Reviews Cancer*.

Attwood, T., Mitchell, A., Gaulton, A., Moulton, G. et Tabernero, L. (2006). The prints protein fingerprint database : functional and evolutionary applications. *Encyclopaedia of Genetics, Genomics, Proteomics and Bioinformatics*.

Bairoch, A. e. a. (2005). The universal protein resource (uniprot), *www.uniprot.org*. *Nucleic Acids Res*, pages 1362–4962.

Barton, G. (2002). Scanps version 2.3.9 user guide. *University of Dundee, UK*.

Bateman, A., Coin, L., Durbin, R., Finn, R. D., Hollich, V., Griffiths-Jones, S., Khanna, A., Marshall, M., Moxon, S., Sonnhammer, E. L. L., Studholme, D. J., C., Y. et Eddy, S. R. (2004). The pfam protein families database. *Nucleic Acids Res, Database Issue*, pages D128–D141.

Bayoudh, S. (2007). Apprentissage par proportion analogique. *Phd thesis, Irisa, Universite de Rennes 1*.

Ben-Gal, I., Shani, A., Gohr, A., Grau, J., Arviv, S., Shmilovici, A., Posch, S. et Grosse, I. (2005). Identification of transcription factor binding sites with variable-order bayesian networks. *Bioinformatics*, pages 21(11) :2657–2666.

Bergadano, F. et Varricchio, S. (1996). Learning behaviors of automata from multiplicity and equivalence queries. *SIAM Journal on Computing*, 25:1268–1280.

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N. et Bourne, P. E. (2008). http ://www.rcsb.org/pdb.

Biermann, A. et Feldman, J. (1972). On the synthesis of finite-states machines from samples of their behavior. Dans *IEEE Trans. Comput.*, pages 592–597.

Boeckmann, B., Bairoch, A., Apweiler, R., Blatter, M. C., Estreicher, A., Gasteiger, E., Martin, M. J., Michoud, K., O'Donovan, C., Phan, I., Pilbout, S. et Schneider, M. (2003). The swiss-prot protein knowledgebase and its supplement trembl in 2003. *Nucleic Acids Res*, 31(1):365–370.

Brazma, A., Jonassen, I., Eidhammer, I. et Gilbert, D. R. (1995). Approaches to the automatic discovery of patterns in biosequences. *Reports in Informatics, Dept. of Informatics, University of Bergen*, 113.

Chomsky, N. (1956). Three models for the description of language. *on Information Theory*.

Coste, F. et Kerbellec, G. (2005). A similar fragments merging approach to learn automata on proteins. Dans *ECML*, pages 522–529.

Coste, F. et Kerbellec, G. (2006). Learning automata on protein sequences. Dans *JOBIM 2006*, Bordeaux.

Coste, F. et Kerbellec, G. (2007). Problème d'optimisation de recherche de cliques pour caractériser des familles de protéines. Dans *ROADEF 2007*, Grenoble.

Coste, F., Kerbellec, G., Idmont, B., Fredouille, D. et Delamarche, C. (2004). Apprentissage d'automates par fusions de paires de fragments significativement similaires et premières expérimentations sur les protéines mip. Dans *JOBIM*, Montréal.

Delepine, L. (2008). http ://webiologie.free.fr.

Denis, F., Lemay, A. et Terlutte, A. (2001). Learning regular languages using rfsa. Dans *Proceedings of the 12th International Conference on Algorithmic Learning Theory, ALT'01*, pages 348–363.

Do, C. B., Mahabhashyam, M. S. P. et Brudno, M. S. B. (2005). Probcons : Probabilistic consistency-based multiple sequence alignment. *Genome Research*, 15:330–340.

Dupont, P., Miclet, L. et Vidal, E. (1994). What is the search space of the regular inference ? *ICGI'94, Grammatical inference and Applications*, pages 25–37. Springer Verlag.

Earl, F., Todd, A., Sillitoe, I., Dibley, M., Redfern, O., Lewis, T., Bennett, C., Marsden, R., Grant, A., Lee, D., Akpor, A., Maibaum, M., Harrison, A., Dallman, T., Reeves, G., Diboun, I., Addou, S., Lise, S., Johnston, C., Sillero, A. et Thornton, J. (2005). The cath domain structure database and related resources gene3d and dhs provide comprehensive domain family information for genome analysis. *Nucleic Acids Research*, 33:D247–D251.

Eddy, S. (1998). Hmmer user's guide : biological sequence analysis using prole hidden markov models. http ://hmmer.wustl.edu/.

Edgar, R. (2004). Muscle : multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res*, 32:1792–1797.

Finn, R. D., Mistry, J., Schuster-Böckler, B., Griffiths-Jones, S., Hollich, V., Lassmann, T., Moxon, S., Marshall, M., Khanna, a., Durbin, R., Eddy, S. R., L., S. E. L. et A., B. (2006). Pfam : clans, web tools and services. *Nucleic Acids Res, Database Issue*, pages D247–D251.

Fredouille, D. (2003). Inférence d'automates finis non déterministes par gestion de l'ambiguïté, en vue d'applications en bioinformatique. *Thèse de doctorat de l'Université de Rennes 1.*

Fu, D., Libson, A., Miercke, L. J., Weitzman, C., Nollert, P., Krucinski, J. et Stroud, R. M. (2000). Structure of a glycerol-conducting channel and the basis for its selectivity. *Science*, pages 290(5491) :481–6.

Fujiyoshi, Y., Mitsuoka, K., de Groot, B. L., Philippsen, A., Grubmuller, H., Agre, P. et Engel, A. (2002). Structure and function of water channels. *Curr Opin Struct Biol.*, pages 12(4) :509–15.

Gish, W. (1996). http ://blast.wustl.edu.

Gold, E. M. (1967). Langage indentification in the limit. *Information and control*, 10(5):447 – 474.

Guermeur, Y. (2007). Svm multiclasses, théorie et applications. *HDR, Loria, Nancy 1.*

Henikoff, S. et Henikoff, J. (1992). Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. USA*, 89:10915–10919.

Henikoff, S., Henikoff, J. G., Alford, W. J. et S., P. (1995). Automated construction and graphical presentation of protein blocks from unaligned sequences. *Gene-COMBIS*, pages 17–26.

Higuera, C., Oncina, J. et Vidal, E. (1996). Identification of dfa : data-dependent versus data-independant algorithms. *Grammatical inference Learning Syntax from Sentences, ICGI'96*, pages 311–325.

Hoang, H. (2006). Rapport de stage. *Symbiose Irisa Rennes.*

Hopcroft, J. et Ullman, J. (1979). Introduction to automata theory, languages, and computation. *N. Reading, MA, Addison-Wesley.*

Hulo, N., Sigrist, C., Le Saux, V., Langendijk-Genevaux, P., Bordoli, L., Gattiker, A., De Castro, E., Bucher, P. et Bairoch, A. (2004). Recent improvements to the PROSITE database. *Nucl. Acids Res.*, 32(90001):D134–137.

Idmont, B. (2002). Mesures de similarité et d'entropie pour l'apprentissage d'automates classifieurs de protéines. *Stage Irisa.*

Jensen, K., Styczynski, M., Rigoutsos, I. et Stephanopoulos, G. (2005). A generic motif discovery algorithm for sequential data. *Bioinformatics.*

Jonassen, I., Helgesen, C. et Higgins, D. G. (1996). Scoring function for pattern discovery programs taking into account sequence diversity. *Reports in Informatics, Dept. of Informatics, University of Bergen*, 116.

Jonassen, I.Collins, J. et Higgins, D. (1995). Finding flexible patterns in unaligned protein sequences. *Protein Science*, 4(8):1587–1595.

Lang, K. J. (1992). Random dfa's can be approximately learned from sparse uniform examples. *5th ACM workshop on Computation Learning Theorie*, pages 45–52.

Lang, K. J., Pearlmutter, B. A. et Price, R. A. (1998a). Abbadingo one, http ://abbadingo.cs.unm.edu.

Lang, K. J., Pearlmutter, B. A. et Price, R. A. (1998b). Results of the abbadingo one dfa learning competition and a new evidence-driven state merging algorithm. *Lecture Notes in Computer Science*, 1433:1–12.

Le Roux, A. (2005). Inférence grammaticale sur des alphabets ordonnés : application à la découverte de motifs dans des familles de protéines. *Thèse de doctorat de l'Université de Rennes 1.*

Lerman, I. et Azé, J. (2004). Indice probabiliste discriminant de vraisemblance du lien pour des données volumineuses. *RNTI-E-1, numéro spécial Mesures de Qualité pour la Fouille des Données, H. Briand, M. Sebag, R. Gras, F. Guillet, CEPA-DUES*, pages 69–94.

Mahé, J. (2007). Rapport de stage. *Symbiose Irisa Rennes.*

Morgenstern, B. (1999). Dialign 2 : improvement of the segment-to-segment approach to multiple sequence alignment. *Bioinformatics*, 15:211–218.

Nevill-Manning, C., Wu, T. et Brutlag, Douglas, L. (1998). Highly specific protein sequence motifs for genome analysis. *PNAS*, 95(11):5865–5871.

Nicolas, J., Durand, P., Ranchy, G., Tempel, S. et Valin, A. (2005). Suffix-tree analyser (stan) : looking for nucleotidic and peptidic patterns in genomes. *Bioinformatics*, 21:4408–4410.

Oncina, J. et Garcia, P. (1992). Inferring regular languages in polynomial update time. *Pattern Recognition and Image Analysis*, pages 49 – 61.

Rigoutsos, I. et Floratos, A. (1998). Combinatorial pattern discovery in biological sequences : the TEIRESIAS algorithm. *Bioinformatics*, 14(1):55–67.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, 14:465–471.

Rossier, J. (2008). http ://bio.espci.fr.

Sakakibara, Brown, Hughey, Mian, Sjolander, Underwood et Haussler (1994). Recent methods for RNA modeling using stochastic context-free grammars. Dans *CPM : 5th Symposium on Combinatorial Pattern Matching.*

Schaffer, A. A., Aravind, L., Madden, T. L., Shavirin, S., Spouge, J. L., Wolf, Y. I., Koonin, E. V. et Altschul, S. F. (2001). Improving the accuracy of psi-blast protein database searches with composition-based statistics and other refinements. *Nucleic Acids Res.*, 29:2994–3005.

Schneider, T. D. et Stephens, R. M. (1990). Sequence logos : A new way to display consensus sequences. *Nucleic Acids Res.*, 18:6097–6100.

Searls, D. (1997). Linguistic approaches to biological sequences. *Comput Appl Biosci*, 13:333–344.

Smith, T. et Waterman, M. (1981). Identification of common molecular subsequences. *Journal of Molecular Biology*.

Stroud, R. M., Mierche, L. J., O'Connel, J., Khademi, S., Lee, J. K., Remis, K., Harries, W., Robles, Y. et Akhavan, D. (2003). Glycerol facilitor glpf and the associated aquaporin family of channels. *Curr Opin Struct Biol.*, pages 13(4) :424–31.

Sui, H., Han, B. G., Lee, J. K., Walian, P. et Jap, B. K. (2001). Structural basis of water-specific transport through the aqp1 water channel. *Nature*, pages 20–27.

Thomas, P. D., Campbell, M. J., Kejariwal, A., Mi, H., Karlak, B., Daverman, R., Diemer, K., Muruganujan, A. et Narechania, A. (2003). Panther : A library of protein families and subfamilies indexed by function. *Protein Informatics, Celera Genomics*.

Thompson, J. D., Higgins, D. G. et Gibson, T. J. (1994). Clustal w : improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matirx choice. *Nucleic Acids Res*, 22(22): 4673–4680.

Venkatarajan, M. et Braun, W. (2001). New quantitative descriptors of amino acids based on multidimensional scaling of a large number of physical-chemical properties. *J Mol Model*, 7:445–453.

Yokomori, T. (1994). Learning non-deterministic finite automata from queries and counterexamples. *Machine Intelligence*, 13:169–189.

Yu, S. (1997). Handbook of formal languages, chapter 2 : Regular languages. *Springer*.

Zdobnov, E. et Apweiler, R. (2001). Interproscan - an integration platform for the signature-recognition methods in interpro. *Bioinformatics*, 17(9):847–848.

Table des figures

1.1 “Adrenergic Receptors, Molecule of the month”, schéma 3D d'une protéine de la PDB (Protein Data Bank) au mois d'avril (Berman et coll., 2008). 16
1.2 tableau récapitulatif des 20 principaux acides aminés représentés dans le code génétique (O et U étant rares), source : wikipedia 17
1.3 Code de traduction génétique (Rossier, 2008) 18
1.4 From gene to protein (source : US department of energy) 19
1.5 Diagramme de Venn des propriétés physico-chimiques des acides aminés (source : wikipedia) 22
1.6 Alignement entre deux séquences de la famille des TNF par utilisation du programme Blast dans sa version Blastp avec les paramètres par défaut 23
1.7 Alignement multiple produit par ClustalW et observé à l'aide de Jalview 24
1.8 The Plan7 architecture (HMM architecture (Eddy, 1998). Les carrés indiquent des états de match. Les losanges indiquent des états d'insertion. Les cercles indiquent les états de délétion. 26
1.9 Le symbole ”x” représente un acide aminé quelconque ; ”x(2)” représente exactement 2 acides aminés ; ”x(1,3)” représente entre 1 et 3 acides aminés ; ” * ” représente de 0 à une infinité d'acides aminés ; ”[ILV]” représente le choix entre les acides aminés I, L ou V. 27
1.10 Exemple de PRATT sur la famille des Zinc Finger 28

2.1 Exemple de NFA. 32
2.2 Illustration de la notion de chemin et d'acceptation. 33
2.3 Espace de recherche pour un automate fini non déterministe. 39

3.1 Partitionnement des positions d'un ensemble de séquences. 48
3.2 Représentation de plusieurs SFP dans un jeu de séquences. 50
3.3 Le jeu de séquences S 52
3.4 Décomposition d'une séquence en fragments 52
3.5 Représentation de plusieurs fragments dans le jeu S 52
3.6 Sommets du Graphe G des fragments 52
3.7 Identification de fragments significativement similaires 53
3.8 Graphe G des fragments impliqués dans des SFP représentés par les arêtes 53

3.9 Représentation de plusieurs Blocs de PLMA. En pratique, notre approche utilise des fragments contigus et de même taille. 54
3.10 Le consensus faible représenté par la transitivité dans un Bloc de PLMA classique (composante connexe) 55
3.11 Le consensus fort représenté par un Bloc de PLMA restreint à une clique 55
3.12 Violation in de la contrainte de Consistance entre les positions 1 et 2 associées à la position 3 et 4.. 57
3.13 En cas d'interférence il faut choisir entre les deux Blocs de PLMA . . . 57

4.1 Identifications des positions associées par le PLMA dans le MCA. 66
4.2 Fusions des transitions et des états relatifs aux positions similaires de plusieurs fragments. 66
4.3 Fusions des états liés par le PLMA dans l'automate. 67
4.4 La fusion d'état sur des automates de Moore se révèle plus pratique que sur des automates de Mealy. On peut ainsi conserver l'information d'une insertion d'un acide aminé, et non créer une boucle ou une répétition de la même position. Ici les position "M" et "L" sont fusionnées. 67
4.5 Automate de protéines avec états de gaps 69
4.6 MCA avec une séquence isolée. 70
4.7 Automate avec présence d'un automate universel dû au court circuit créé par la fusion en gap d'une séquence exceptionnelle par rapport au reste de la famille. 70
4.8 Identification d'une exception et fusion des zones de gap. 71
4.9 MCA et mise en valeur des zones similaires. 72
4.10 Automate résultat, exhibant les zones caractéristiques, les chemins exceptions à ces zones, ainsi que les zones de gaps reliant les zones caractéristiques. 73

5.1 On utilise un jeu de séquences d'apprentissage (ici le clan PFAM connu sous le nom de "Viral nucleic acid binding"). 85
5.2 On obtient plusieurs automates en fonction de différents quorum 86
5.3 Pour un quorum de 100%, nous avons un automate équivalent à un motif de type Prosite. 86
5.4 En descendant en quorum, nous découvrons des blocs contenant moins de séquences, ainsi que des exceptions à ces blocs. 87
5.5 Enfin, la vue "plma" nous permet, pour des quorums encore plus petits, d'observer le comportement des familles de séquences. 87
5.6 Pour chaque automate ou tout simplement pour celui qui montre le meilleur score MDL, nous pouvons scanner cet automate sur une base de donnée. 88
5.7 Voici le résultat d'un scan, avec les séquences valides et leur score de parsing. 88

6.1 Distance de chaque séquence de validation pour son acceptation dans l'automate produit en utilisant l'heuristique d'implication. Leave-One-Out produit dans le but d'évaluer la caractérisation de $W+$ avec $W-$ comme contre exemple. Le jeu C est un jeu de contrôle (non MIP), seule sa séquence la plus proche est représentée. 98
6.2 Distribution pour 3 heuristiques du pourcentage de séquences acceptées par un automate de taille approximativement 100 transitions. 100
6.3 Impact du quorum sur la F-mesure. 102
6.4 Consensus Fort, Taille des fragments entre 0 et 40, Seuil Dialign à 1 . . 105
6.5 Consensus Fort, Taille des fragments entre 0 et 40, Seuil Dialign à 5 . . 105
6.6 Consensus Fort, Taille des fragments entre 0 et 40, Seuil Dialign à 10 . . 106
6.7 Consensus Faible, Taille des fragments entre 0 et 40, Seuil Dialign à 1 . 106
6.8 Consensus Faible, Taille des fragments entre 0 et 40, Seuil Dialign à 5 . 107
6.9 Consensus Faible, Taille des fragments entre 0 et 40, Seuil Dialign à 10 . 107
6.10 Consensus Fort, Taille des fragments entre 0 et 15, Seuil Dialign à 1 . . 108
6.11 Consensus Fort, Taille des fragments entre 0 et 15, Seuil Dialign à 5 . . 108
6.12 Consensus Fort, Taille des fragments entre 0 et 15, Seuil Dialign à 10 . . 109
6.13 Consensus Faible, Taille des fragments entre 0 et 15, Seuil Dialign à 1 . 109
6.14 Consensus Faible, Taille des fragments entre 0 et 15, Seuil Dialign à 5 . 110
6.15 Consensus Faible, Taille des fragments entre 0 et 15, Seuil Dialign à 10 . 110
6.16 Consensus Fort, Taille des fragments entre 0 et 10, Seuil Dialign à 1 . . 111
6.17 Consensus Fort, Taille des fragments entre 0 et 10, Seuil Dialign à 5 . . 111
6.18 Consensus Fort, Taille des fragments entre 0 et 10, Seuil Dialign à 10 . . 112
6.19 Consensus Faible, Taille des fragments entre 0 et 10, Seuil Dialign à 1 . 112
6.20 Consensus Faible, Taille des fragments entre 0 et 10, Seuil Dialign à 5 . 113
6.21 Consensus Faible, Taille des fragments entre 0 et 10, Seuil Dialign à 10 . 113
6.22 Consensus Fort, Taille des fragments entre 10 et 30, Seuil Dialign à 1 . . 114
6.23 Consensus Fort, Taille des fragments entre 10 et 30, Seuil Dialign à 5 . . 114
6.24 Consensus Fort, Taille des fragments entre 10 et 30, Seuil Dialign à 10 . 115
6.25 Consensus Faible, Taille des fragments entre 10 et 30, Seuil Dialign à 1 . 115
6.26 Consensus Faible, Taille des fragments entre 10 et 30, Seuil Dialign à 5 . 116
6.27 Consensus Faible, Taille des fragments entre 10 et 30, Seuil Dialign à 10 116
6.28 Courbe MDL 117
6.29 Classes de langages et séquences biologiques (Searls, 1997). 122

Liste des algorithmes

1 Principe des algorithmes gloutons par fusion à partir d'un ensemble $S+$ et d'un ensemble de contre-exemples $S-$ 39
2 Schéma général de la phase d'alignement 59
3 Algorithme de caractérisation par méthode Strictement SFP 60
4 Algorithme de caractérisation par méthode en Clique Semi-Exhaustive . 61
5 Algorithme de caractérisation par méthode de Graines de Bloc de PLMA 62
6 Approche globale de notre approche d'apprentissage de NFA incluant la fusion du PLMA et les zones de Gap. 69
7 Approche par fusion d'états et identification des chemins exceptions . . 72
8 Identification et extension aux propriétés physico-chimiques 74
9 Approche complète incluant la fusion du PLMA et les zones de Gap, les Exceptions, l'identification des groupes physico-chimiques ainsi que l'extension aux groupes physico-chimiques. 75

Résumé

Cette thèse propose une nouvelle approche de découverte de signatures de familles de protéines. Etant donné un échantillon (non-aligné) de séquences appartenant à une famille structurelle ou fonctionnelle de protéines, cette approche infère des automates finis non-déterministes (NFA) caractérisant la famille.
Un nouveau type d'alignement multiple nommé PLMA est introduit afin de mettre en valeur les similarités partielles et locales significativement similaires. A partir de ces informations, les modèles de type NFA sont produits par un procédé relevant du domaine de l'inférence grammaticale. Les modèles NFA, présentés ici sous le nom de Protomates, sont des modèles graphiques discrets de forte expressivité, ce qui les distingue des modèles statistiques de type profils HMM ou des motifs de type Prosite.
Les expériences menées sur différentes familles biologiques dont les MIP et les TNF, montrent un succès sur des données réelles.

Mots-clés

bioinformatique, inférence, grammaticale, famille, protéine, automate, alignement, multiple.

Abstract

This thesis shows a new approach out of discovering protein families signatures. Given a sample of (unaligned) sequences belonging to a structural or functional family of proteins, this approach infers non-deterministic automata characterizing the family.
A new kind of multiple alignment called PLMA is introduced in order to emphasize the partial and local significant similarities. Given this information, the NFA models are produced by a process stemming from the domain of grammatical inference. The NFA models, presented here under the name of Protomata, are discreet graphical models of strong expressiveness, which distinguishes them from statistical models such as HMM profiles or pattern models like Prosite patterns.
The experiments led on various biological families, among which the MIP and the TNF, show a success on real data.

Keywords

bioinformatics, grammatical, inference, protein, family, automata, multiple, alignment.